A FLEXIBLE SOFTWARE DEFINED RADIO PLATFORM FOR WAVEFORM DEVELOPMENT

Michael Moore (Southwest Research Institute, San Antonio, TX, msmoore@swri.org)
Mike Pilcher (Southwest Research Institute, San Antonio, TX, mpilcher@swri.org)
Max Robert (Virginia Tech MPRG, Blacksburg, VA, probert@vt.edu)

ABSTRACT

This paper describes the *SDR on a CD* system, a flexible Software Defined Radio (SDR) platform developed at Southwest Research Institute® that is an ideal test-bed for prototyping waveform software and experimenting with waveform design, as well as for learning about the Software Communications Architecture (SCA). The system uses ordinary Personal Computers running Linux and the open-source SCA implementation from the Virginia Tech Mobile and Portable Radio Research Group (MPRG) called OSSIE to create a cheap and flexible platform for base-band processing. The system includes USB-based transmitter and receiver hardware modules constructed from Commercial-Off-the-Shelf (COTS) components to implement the digital base-band to radio frequency interfaces. A bootable compact disk contains the complete waveform software. *SDR on a CD* was developed to help address two difficulties with developing SDR applications: 1) the steep learning curve for new engineers to become proficient in developing SCA waveform software and 2) the sizeable gap in environments and tools for transitioning between waveform functional simulations in tools such as SIMULINK® and MATLAB® and implementation on real radio platforms. The system has been used both for training, and for creating working radio prototypes.

1. INTRODUCTION

Software Defined Radio (SDR) technologies have been enjoying a period of rapid growth in recent years, as evidenced by the existence of organizations such as the SDR Forum and the Object Management Group's (OMG) Software Based Communications Special Interest Group (SBC PSIG). The United States Government Department of Defense (DoD), which depends heavily on wireless communications as a core technology for current and future combat activities, has a vested interest in the creation and adoption of SDR architectural standards. This is to promote interoperability of radios, reduce the dependence of radio applications on the underlying platform, provide paths to cost-effectively upgrade radio technologies in the future, and reduce long-term costs. The DoD initiative called the Joint Tactical Radio System (JTRS [1]) project has developed and adopted a standard radio architecture called the Software Communications Architecture (SCA). This standard is based on other standards, such as the Common Object Request Broker Architecture (CORBA) and Portable Operating System Interface to Unix (POSIX). SCA defines a set of base services referred to as the SCA Core Framework (CF), which include eXtensible Markup Language (XML) parsing and SCA Domain Management (DM) services that are used to configure an SCA radio. The operating system, the middleware, and the CF services together are referred to as the SCA Operating Environment (OE).

Although the SCA architecture has been formally adopted by JTRS, it continues to evolve. There are many technical challenges and issues that are still being researched, and these have proven costly in the DoD efforts to develop and procure SCA radios.

One of the difficulties of SDR is that it covers a wide range of disciplines, and hence can be overwhelming to the entry-level engineer. SDR development extends the skill set necessary for traditional wireless system development, adding component-based design and management to the suite of more traditional technologies such as embedded signal processing development and overall system design.

The SCA itself can be difficult to learn. It leverages several existing technologies, CORBA and XML, with which developers may not necessarily be familiar; and it creates a new architecture, the details of which can be complicated. In order to ease this transition, developers at Virginia Tech created OSSIE (Open Source SCA Implementation::Embedded)[2]. The goal of OSSIE is to provide a free, open-source, C++ implementation of the SCA that is relatively simple, easing the new developer's learning curve.

Another challenge to be faced when transitioning away from traditional fixed solutions to SDR, particularly to the software-centric SCA architecture, is that there are not yet sufficient tools and accepted best practices for how to start with a functional simulation and proceed in an incremental fashion toward an SCA implementation on the embedded target. Functional simulations are commonly conducted using SIMULINK® and MATLAB®. With these tools, the waveform functional behavior can be developed and refined in a largely implementation-independent fashion. Code generation tools for SIMULINK® exist but not for generating SCA implementations directly onto custom embedded platforms. The implementation on the target platform is most often completed manually, after the functional design has been simulated. Simulations, however, are usually tested with synthetic interfaces; not with actual wireless channels or radio frequency (RF) to digital interfaces. This can mean that the functional design has not taken into account the impacts of having a channel with non-ideal properties such as non-gaussian noise, complex sets of interference sources, multi-path effects, etc. What is needed to aid in this transition is a method of rapidly prototyping waveforms and testing the functional design against the effects of a real RF channel. This would require an inexpensive and easy-to-use testbed with a simple SCA operating environment and a physical channel for quickly and inexpensively prototyping functional designs, and testing against a real physical channel, before transitioning into the implementation phase.

Southwest Research Institute® (SwRI®) has created a flexible software defined radio platform for waveform development that provides some support for easing these difficulties. The *SDR on a CD* system described in this paper bundles a freely available operating system (Linux) with open-source implementations of CORBA and an SCA core framework on a bootable CD. The system includes USB-based modules that interface between baseband and radio frequency subsystems and provide a simple and flexible narrow-band physical channel. The result is a very low-cost testbed for investigating and experimenting with these difficult technical issues. Although the *SDR on a CD* platform is a fully functional SDR, it is not meant to be a military or commercial grade radio system. Rather, it is a research platform on which trade studies and proof-of-concept experiments can be performed. We call this system, described in the following section, the *SDR on a CD* because all of the waveform software is on a bootable CD.

2. THE SDR ON A CD SYSTEM

The *SDR on a CD* platform consists of four major components: the Operating System (OS) and software framework, the waveform software implementation, a computing platform, and the RF backend hardware. These four components of the system are analogous to a Commercial Off-the-Shelf (COTS) SDR system. In order to provide the flexibility necessary for a research platform, these components are loosely coupled. This coupling is implemented with a standard architecture and a standard software framework. This allows the *SDR on a CD* platform to not only be easily reconfigured, but also to easily integrate with existing SDR system components where necessary.

The *SDR on a CD* system utilizes commodity desktop processing hardware (desktop or laptop personal computers) for base-band processing and provides USB-based transmitter and receiver hardware modules that implement the radio frequency interfaces. A bootable compact disk contains all of the necessary waveform software, including the operating system (Linux), the middleware (ACE/TAO CORBA [3]), the Software Communications Architecture Core Framework (OSSIE from Virginia Tech MPRG [2]), and the waveform processing software (FSK, PSK, MSK, or other). The waveform processing software is highly extensible and, together with the narrow-band physical channel provided by the universal serial bus (USB) modules, enables rapid prototyping and experimentation with SDR algorithms. The *SDR on a CD* system follows the "software first" design approach and could help bridge the gap that exists between functional design activities, such as simulation and prototyping and implementation on specific SCA platforms.

The system use-case involves choosing a laptop with the basic system requirements and using the CD on that laptop to boot the SDR system (see Figure 1).

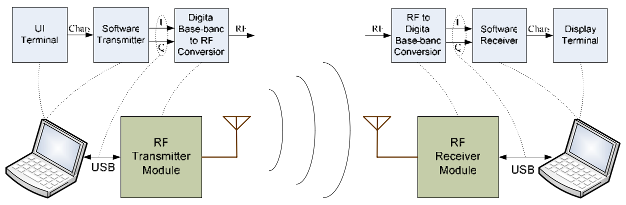


Figure 1. The SDR on a CD System Concept

2.1. OS and Middleware

The base component in the *SDR on a CD* platform is the operating system and the middleware. The OS for our platform is Linux, specifically the Knoppix LiveCD variant of the Debian Linux distribution. This LiveCD system was chosen in order to provide the flexibility necessary to run the platform on most modern laptops. The Linux LiveCD framework takes care of all the hardware detection and configuration, as well as packaging and initializing the software framework. This LiveCD deployment allows the SwRI *SDR on a CD* to run on virtually any laptop with a USB port with no additional development effort.

In order for the waveforms of the *SDR on a CD* to remain as platform-independent as possible, the SCA OSSIE core framework is used. This core framework allows the SDR to reconfigure quickly and cleanly and provides for a componentized architecture. Thus, the multiple modulators and demodulators in the SDR system do not have to know the overall system architecture and can be easily coded and tested independently.

2.1.1 OSSIE

The OSSIE implementation of the SCA attempts to strike a compromise between simplicity and functionality. As the functionality supported by a specific framework grows, its complexity increases. Thus, the concept behind OSSIE is to keep the initial complexity of the framework to a relatively low level, providing a relatively simple starting point. In the case of OSSIE, the framework supports the management (i.e., loading, unloading, starting, stopping, and other such functions) of waveforms. Beyond this, there are other features that OSSIE does not immediately support such as automatic matching of binary to hardware and multiple implementations of components.

The open nature of the framework then allows the specific developer to enhance the framework with whatever additional functionality is needed for his specific purposes. For example, in the case of a cognitive engine, the developer may want to extend the device assignment sequence to perform some specialized optimization. Other possibilities include the modification of the framework to include active power management strategies. The relative openness of the implementation allows OSSIE to be integrated into research, providing the research team with the ability to investigate several "what-if" scenarios without necessarily being bound by the current SCA specifications.

2.1.2 The OSSIE Platform

OSSIE is not stand-alone software; it relies on other existing packages to provide necessary additional functionality. Because OSSIE is open-source, it was deemed important that any additional dependencies also be open-source. The additional dependencies include OmniORB (CORBA implementation) [4], OmniEvent (Event service for OmniORB) [5], and Xerces (XML parser) [6]. An additional package that has proven useful for desktop system component development is ACE [7], which greatly simplifies thread management and data exchange.

OSSIE is designed to reduce the problems that an entry-level engineer may encounter. One of these problems is lack of familiarity with tool chains. Thus, it was deemed important to cover operating systems that developers may use. As of the publication of this paper, OSSIE has been successfully tested in Linux (Fedora Core 3 and 4, Suse) and Windows XP/2000. OSSIE has also been shown to work with OS-X, but that operating system is not directly supported yet. The above-mentioned operating systems are aimed at the desktop computer market. Beyond desktop computers, OSSIE has also been shown to work on the ARM core running Linux on an OMAP 5912, thus extending OSSIE into the embedded environment. An Open Embedded package is available that automatically builds the image necessary to run OSSIE on the OMAP processor. OSSIE is available for download under the LGPL (Lesser General Public License) from http://ossie.mprg.org.

2.2. Waveform Software

The waveforms used on the *SDR on a CD* are fully implemented in software. The current platform contains modulator and demodulator implementations of Frequency Shift Keyed (FSK), Phase Shift Keyed (PSK), and Minimum Shift Keyed (MSK) modulation schemes. These waveform components are implemented in standard C++ and interface with the system using the OSSIE core framework. Each block in the waveform is implemented as a standalone SCA Component. Thus, blocks are easily interchangeable; and a block used in one encoding/modulation scheme can be used in another scheme with a minimum of development effort.

The block diagram in Figure 2 illustrates the flow of the transmitter waveforms of the *SDR on a CD*. The generic transmitter consists of an ASCII to Binary conversion component, a framing and coding component, a differential encoding component, and a modulator component. Each of these components can be enabled or disabled as needed for a particular waveform. Specific parameters for each component can be set at runtime via the domain profile configuration supported by the OSSIE core framework. This runtime configuration allows the user to change settings in a well-defined XML file format without the need to recompile the waveform software. An example of this runtime reconfigurability is the ability to adjust the coding scheme in the framing and coding block.

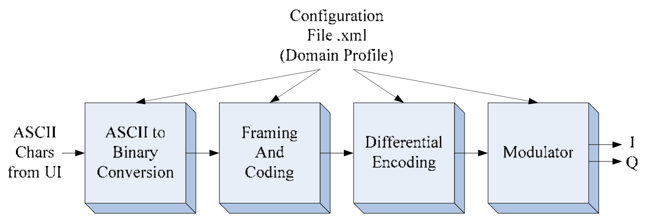


Figure 2. Transmitter Software Block Diagram

Figure 3 depicts the generic receiver software waveform block diagram. As with the transmitter waveform software, the component blocks are configurable at runtime. To create a flexible demonstration waveform with which to test the *SDR on a CD* system, we implemented a differential-based demodulator that demodulates FSK, MSK, and PSK signals.

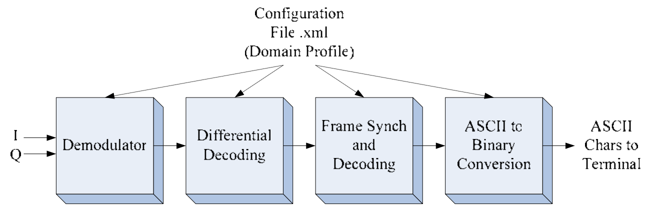


Figure 3. Receiver Software Block Diagram

2.3. RF Hardware

The RF hardware that implements the *SDR on a CD* physical channel consists entirely of COTS components. This design decision was made in order to reduce the cost of the RF system, as well as reduce the overall system complexity. Both the transmitter and receiver interface with the rest of the SDR system via USB.

The block diagram in Figure 4 illustrates the hardware components used to implement the transmitter RF module. The initial block in the RF module is a USB hub that allows communication with the iMic and the Labjack components. The iMic is a USB sound card that is used as the Digital to Analog (D/A) converter for the system. The Labjack is a USB-to-Parallel converter that controls the programming of the Phase-Lock-Loop (PLL) synthesizer board. The PLL works in conjunction with the up converter to move the baseband data up to the target RF frequency of 450 MHz. Clearly, the decision to use a sound card for the D/A interfaces restricts the waveform bandwidth. However, the demonstration scenario for this system was narrow-band waveforms. With fairly simple modifications, a wider band equivalent of the transmitter could be built without requiring any changes to the basic architecture.

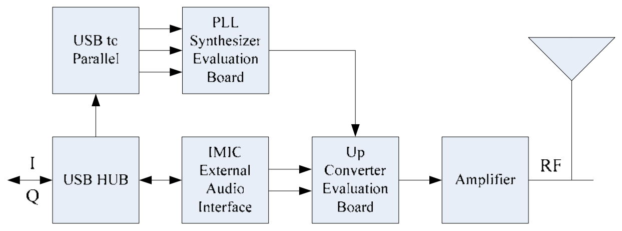

USB to Parallel
PLL Synthesizer Evaluation Board
I
Q
USB HUB
IMIC External Audio Interface
Up Converter Evaluation Board
Amplifier
RF

Figure 4. Transmitter RF Module Block Diagram

The block diagram in Figure 5 shows the receiver RF module components. The receiver, like the transmitter, has an iMic, a Labjack, and a PLL synthesizer. The iMic is the Analog to Digital (A/D) converter for the system. As with the transmitter, the Labjack controls the programming of the PLL. The PLL controls the frequency of the down converter.

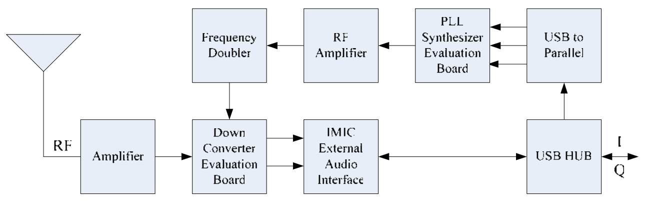

Frequency Doubler
RF Amplifier
PLL Synthesizer Evaluation Board
USB to Parallel
RF
Amplifier
Down Converter Evaluation Board
IMIC External Audio Interface
USB HUB
I
Q

Figure 5. Receiver RF Module Block Diagram

3. APPLICATIONS

The *SDR on a CD* system is intended to be an inexpensive COTS SDR platform for prototyping narrow-band waveforms. It is suitable for learning about communications and the SCA, for research, and for proof-of-concept demonstrations. The platform has multiple possible uses, such as waveform algorithm evaluation, experimenting with SCA optimization techniques (e.g., improving boot and/or configuration time and evaluating design and implementation tradeoffs against varying quality physical channels.

3.1. Waveform Applications

The *SDR on a CD* can be used to test and evaluate new and existing SCA waveform implementations. These waveforms can be tested with or without the RF hardware depending on the maturity of the waveform. The waveform can be tested in a short circuit manner wherein the modulator is hooked directly to the demodulator. This setup can be used to verify that the system works correctly without the RF hardware.

The waveforms under test can also be evaluated for system reconfigurability. Multiple waveforms can be activated simultaneously to determine performance issues, system concurrency issues, or any other system integration level issues. In addition to concurrent waveform execution, the waveforms can be evaluated for system-level timing information to aid in scheduling analysis or platform selection.

3.2. System Applications

The *SDR on a CD* system can be used for development of system-level algorithms and techniques. For instance, a current issue plaguing the SCA development effort is boot time. The amount of time from when the system is started to when it is ready for operation is critical in many applications. Another critical timing parameter is the amount of time needed to reconfigure the SDR system. Different techniques for reducing boot or reconfiguration time can be implemented and tested easily using the *SDR on a CD* system.

The *SDR on a CD* system could also be modified to use a different core framework or CORBA implementation to explore platform middleware tradeoffs.

4. CONCLUSION

The *SDR on a CD* system created at SwRI allows the SDR team at SwRI to quickly and easily experiment with SDR design and implementation tradeoffs. This flexible platform is invaluable in the initial implementation efforts for the SCA and JTRS framework to perform tradeoff and proof-of-concept studies.

The waveform processing software is highly extensible, and together with the narrow-band physical channel, provided by the USB modules, enables rapid prototyping and experimentation with SDR algorithms.

The *SDR on a CD* system follows the "software first" design approach and could help bridge the gap that exists between functional design activities, such as simulation and prototyping, and implementation on specific SCA platforms. Another important use of the *SDR on a CD* platform is for helping engineers understand and learn to use the SCA. The *SDR on a CD* system will help smooth the transition for new engineers and those unfamiliar with SCA into developing SCA waveform software.

The easily extensible waveform processing software, together with the narrow-band physical channel provided by the USB modules, enables rapid prototyping and experimentation with SDR algorithms. Engineers will have to balance a very large number of tradeoffs as SDR technology becomes standard. A prototyping platform that can measure both the resource costs and demonstrate effectiveness of various FEC codes, adaptive equalization techniques, and other radio functions will assist engineers in making the most optimal design decisions for software defined radios.

REFERENCES

[1] JTRS Joint Program Office, "Joint Tactical Radio System – JTRS," http://jtrs.army.mil/sections/technicalinformation/fset_technical_sca.html.

[2] MPRG at Virginia Tech, "Open Source SCA Implementation::Embedded (OSSIE) Site," http://www.mprg.org/research/ossie/index.html.

[3] D.C. Schmidt et al., "TAO: A High Performance Endsystem Architecture for Real-Time CORBA", *IEEE Communications Magazine*, 14(2), Feb. 1997.

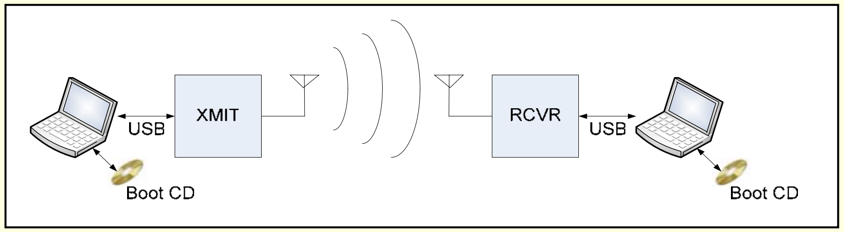
[4] http://omniorb.sourceforge.net/

[5] http://www.omnievents.org/

[6] http://xml.apache.org/

[7] http://www.cs.wustl.edu/~schmidt/ACE.html

SDR on a CD



A FLEXIBLE SOFTWARE DEFINED RADIO PLATFORM FOR WAVEFORM DEVELOPMENT

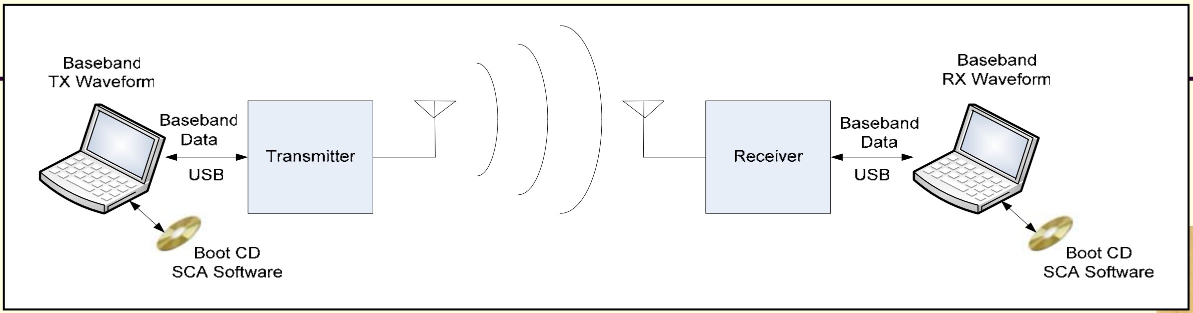
Michael Moore (msmoore@swri.org)
Mike Pilcher (mpilcher@swri.org)
Southwest Research Institute, San Antonio, TX
(http://www.swri.org/4org/d10/comm/commeng/softrad.htm)

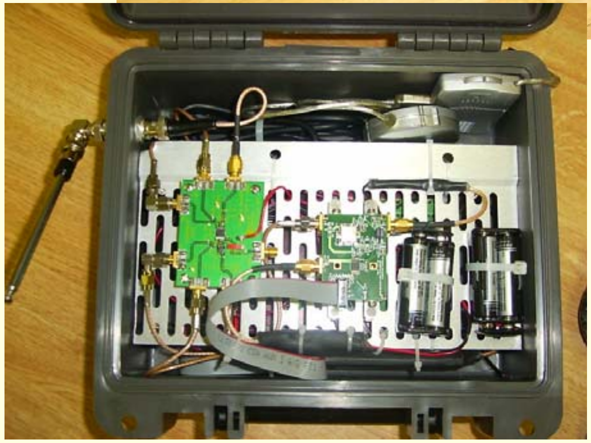
Max Robert (probert@vt.edu)
Virginia Tech MPRG, Blacksburg, VA

SDR Internal Research Project

- Objectives
  - Examine JTRS / SCA architecture
  - Identify areas in need of additional innovation
  - Demonstrate a fully functional SCA SDR
- Build a demonstration SDR platform
  - Demonstrate SDR benefits to a wide audience
  - Perform functional design trades on a real channel
  - Experiment with SCA architecture / perform trades
  - Test-bed for solutions to difficult problems

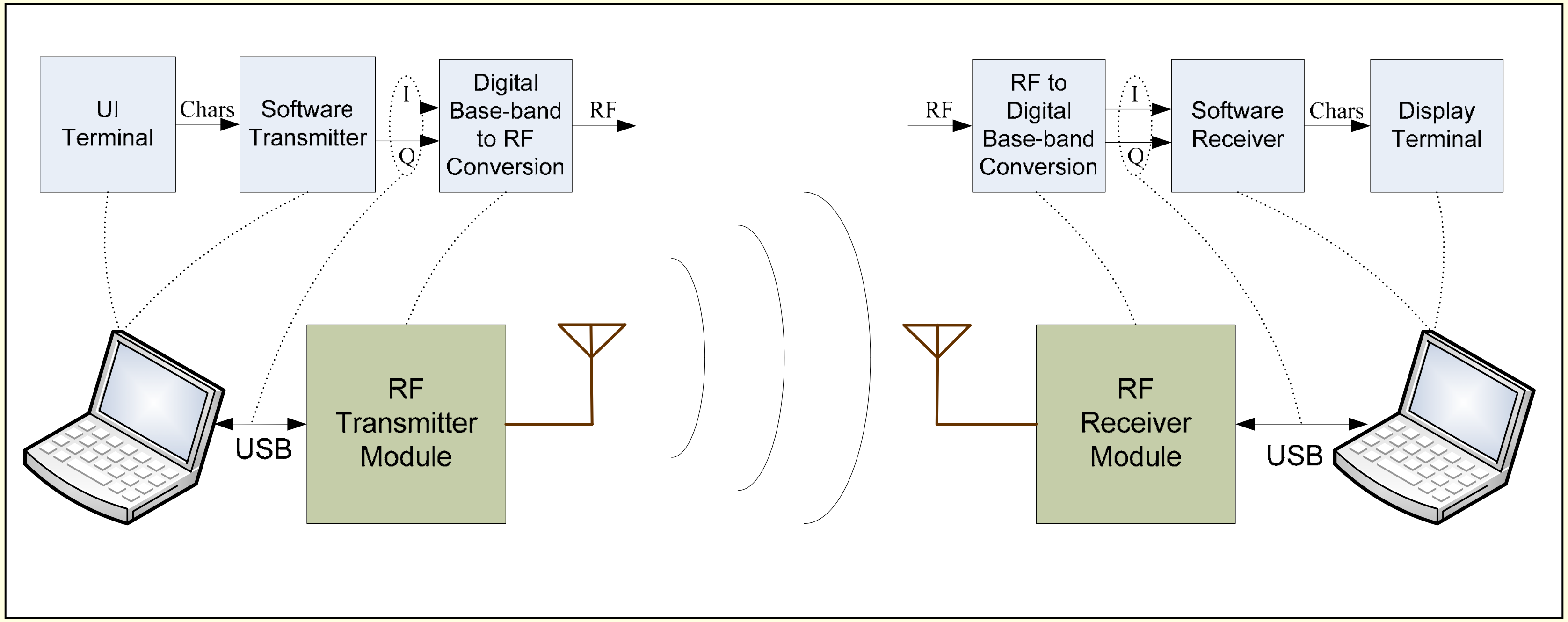
SwRI "SDR on a CD" System



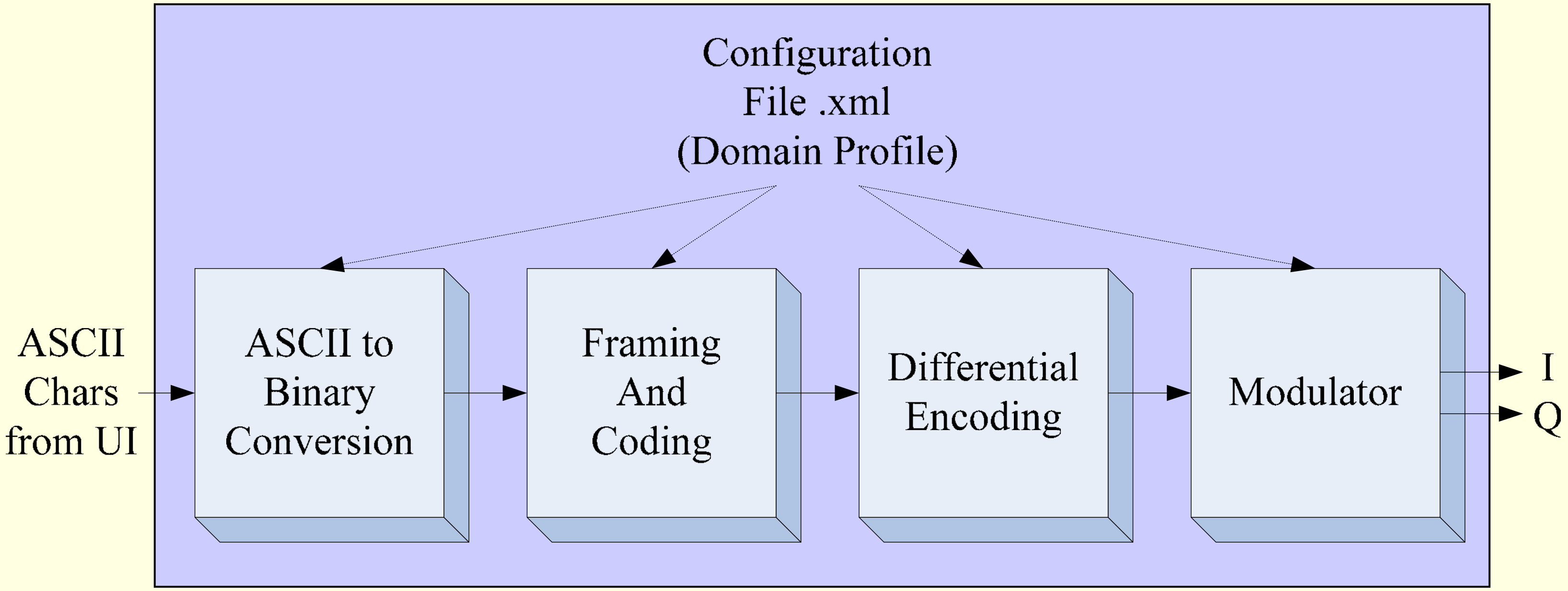


- "SDR on a CD" system developed at SwRI
  - FSK/PSK/MSK modem
  - Base-band on standard laptops
  - Bootable SDR CD
  - SCA Operating Environment
- USB-based RF interface devices
- Functional, integrated SDR
- Useful for
  - Learning about SDR and SCA
  - Demonstrating benefits of SDR
  - Performing functional trades
  - Designing waveform software

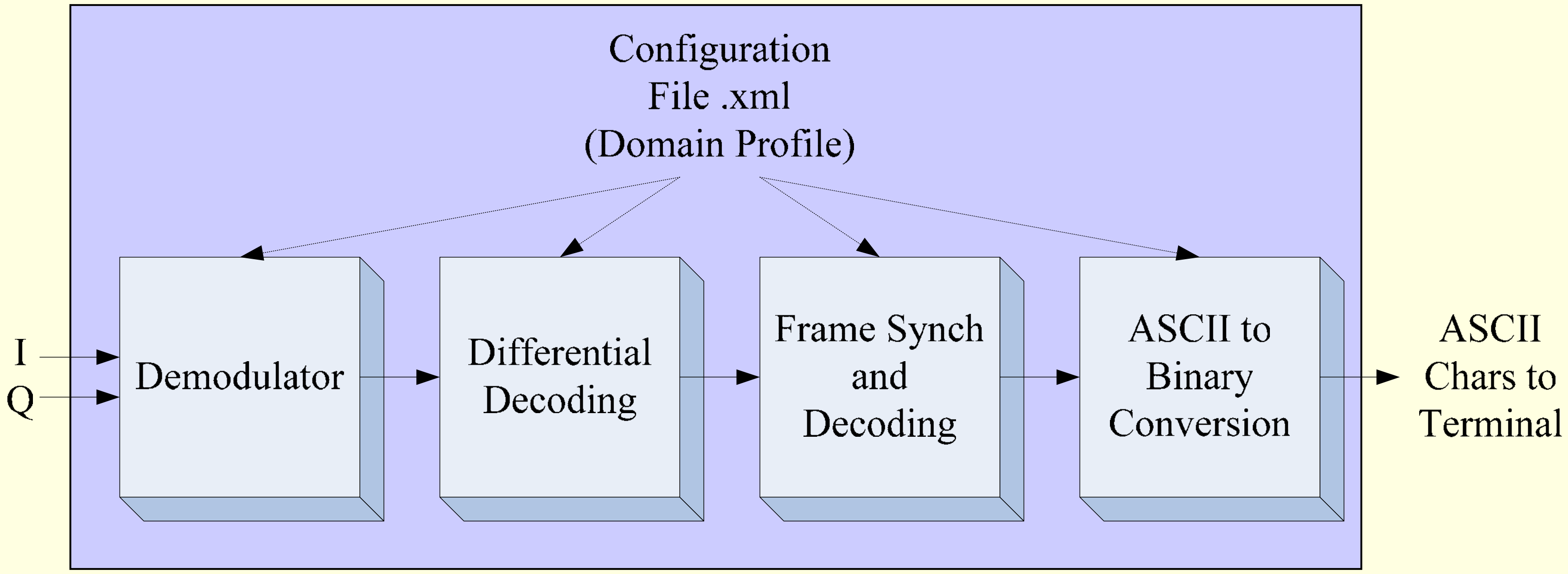
SDR on a CD Concept



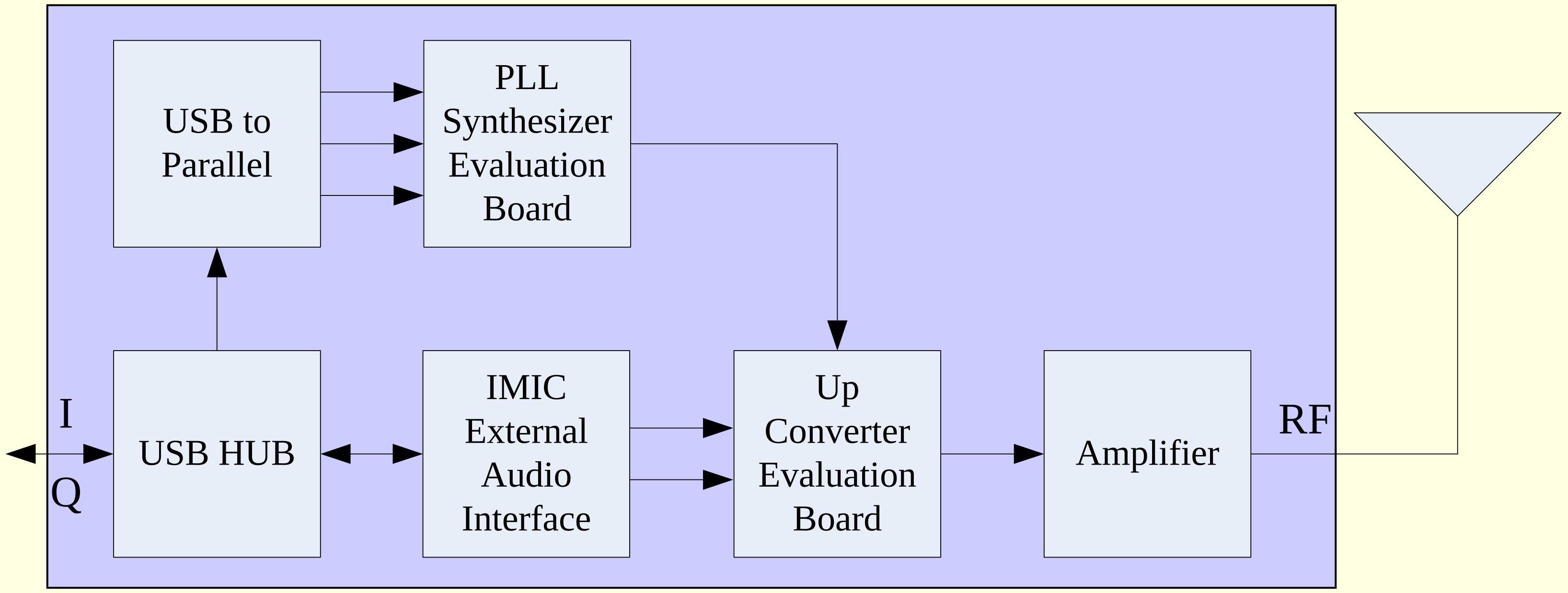
Transmitter Software



Receiver Software

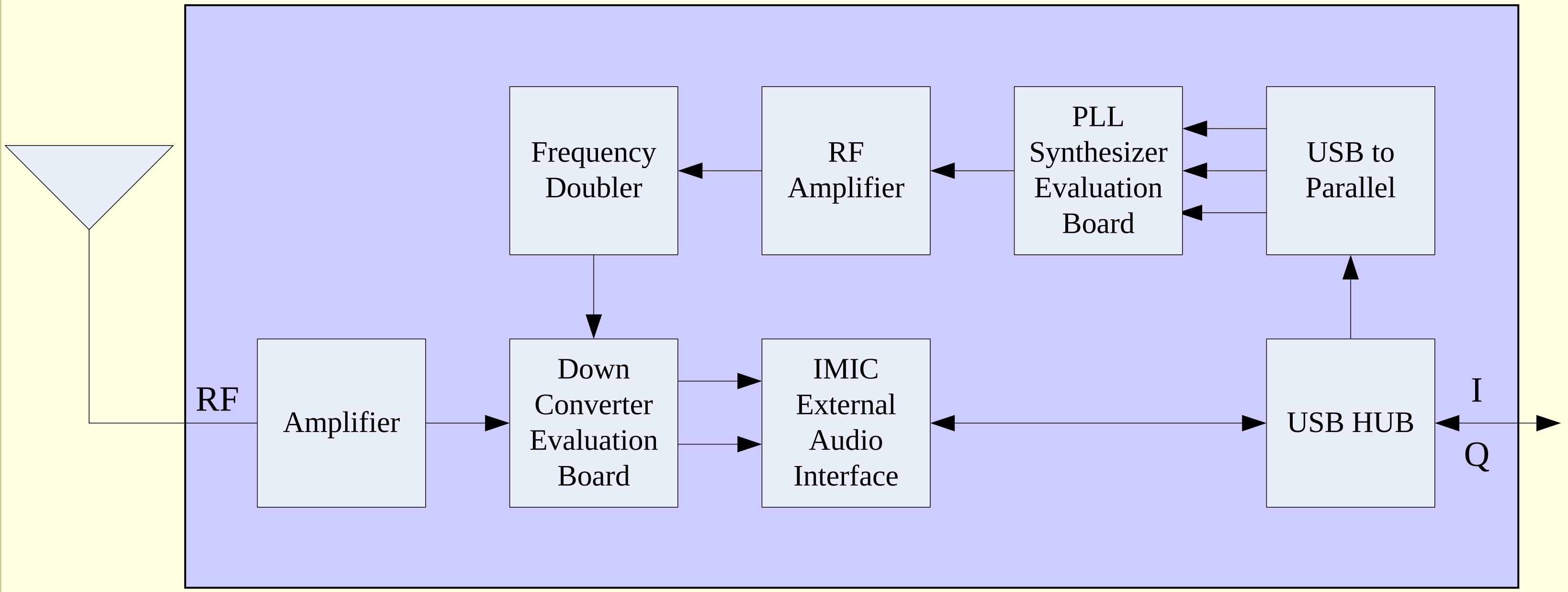


RF Transmitter Module


USB to Parallel
PLL Synthesizer Evaluation Board
I
Q
USB HUB
IMIC External Audio Interface
Up Converter Evaluation Board
Amplifier
RF

- COTS Components
- VHF Operation
- Connects to Laptop with standard USB
- Battery Powered

RF Receiver Module



Frequency Doubler
RF Amplifier
PLL Synthesizer Evaluation Board
USB to Parallel
RF
Amplifier
Down Converter Evaluation Board
IMIC External Audio Interface
USB HUB
I
Q

- COTS Components
- VHF Operation
- Connects to Laptop with standard USB
- Battery Powered

Conclusions & Research Direction

- SDR on a CD is a real radio and has proven useful for SDR research
- Future work
  - SDR engineering / prototyping
    - Improve / optimize framework (resources, boot time)
    - Demonstrate benefits of reconfigurable SDR (space)
    - Prototype waveforms for space applications (NASA)
  - SDR research
    - Adaptive spectrum management (cognitive radio)
    - Smart antennas (both directional and band flexible)
    - Digital Pre-Distortion (DPD)
  - SDR tool kit (model-driven development tools)