SDR TESTBENCH FOR SATELLITE COMMUNICATIONS

Kris Huber (Array Systems Computing Inc., Toronto, Ontario, Canada,
khuber@array.ca);
Weixiong Lin (Array Systems Computing Inc., Toronto, Ontario, Canada).

ABSTRACT

A Software Defined Radio (SDR) test-bed designed to demonstrate the regenerative capability of onboard processing for satellites was developed as a proof-of-concept system. The aim of this project was to perform an analysis of system capabilities subject to the underlying requirement that the prototype platform was composed of generic commercial off-the-shelf (COTS) hardware and software components and be compliant to the Software Communications Architecture (SCA) version 2.2. The system performs demodulation and decoding for the second generation digital video broadcasting standard for satellite (DVB-S2). The technical analysis performed was designed to demonstrate and validate the themes of interoperability and upgradeability. Performance results demonstrated the prototype's ability to achieve these aims; however, overall system performance (such as throughput) was less than anticipated primarily due to the overhead of software overhead management issues.

1. INTRODUCTION

The multimedia/communications architectures for next generation satellite networks will aim at providing faster speed and superior quality of service (QoS) over the conventional transparent satellite architectures of today. Meeting quality of service implies increasing reliability of current systems and expanding/developing new systems and services to further address the technical and business demands of today's customers. To achieve these goals, the satellites supporting the network must be able to perform baseband onboard regenerative processing (OBRP) and exhibit the capabilities of interoperability and upgradeability [1].

In an initial response to these issues, an SCA [2] compliant SDR test-bed for demonstrating the regenerative capability of onboard processing was developed as a proof-of-concept system. SDR technology which is the process of using software to functionally define and control reconfigurable hardware devices, is a maturing technology. The SCA forces both the radio and waveforms deployed on the radio to conform to an established set of standards (referred to as the core-framework). The SCA (or some variant thereof) is widely expected to play a prominent role in the wireless landscape for next-generation (NGN) terrestrial systems – and thus by extension, NGN satellite networks will necessarily have to become interoperable with the terrestrial networks in order to maintain relevance. For this reason, the SCA was also chosen as the control software for this radio test-bed.

Two waveforms were chosen to be prototyped, namely DVB-S [3] and selected components of DVB-S2 [4]. It was planned that the OBRP would be able to simulate various scenarios such as remotely upgrading the system from utilizing a DVB-S waveform to a DVB-S2 waveform; or to provide reception using one format and then perform baseband switching and provide transmission on another format. The COTS components used were a simple dual Opteron desktop PC in which a PCI radio card was inserted. The operating system used Linux Core 2, on which the SCA version 2.2 was configured.

The remainder of this paper is as follows: Section 2 discusses the COTS components used for the development of the radio platform, while Section 3 outlines the waveform development process. Section 4 discusses the testing and validation setup, and Section 5 summarizes some of the results.

2. PROCESSOR ARCHITECTURE

From the implementation side, the project goal from the outset was to develop a standardized OBRP for the deployment of the DVB-S and DVB-S2 waveforms using COTS elements. It was also deemed desirable to perform as much of the signal processing as possible on general purpose processors (GPPs), while still retaining the use of FPGAs and ASICs where bandwidth requirements dictated. The SCA was chosen to manage the operating environment (OE) since it forces standardization, and it promotes functionalities such as upgradeability and interoperability of all the software components contained within the radio.

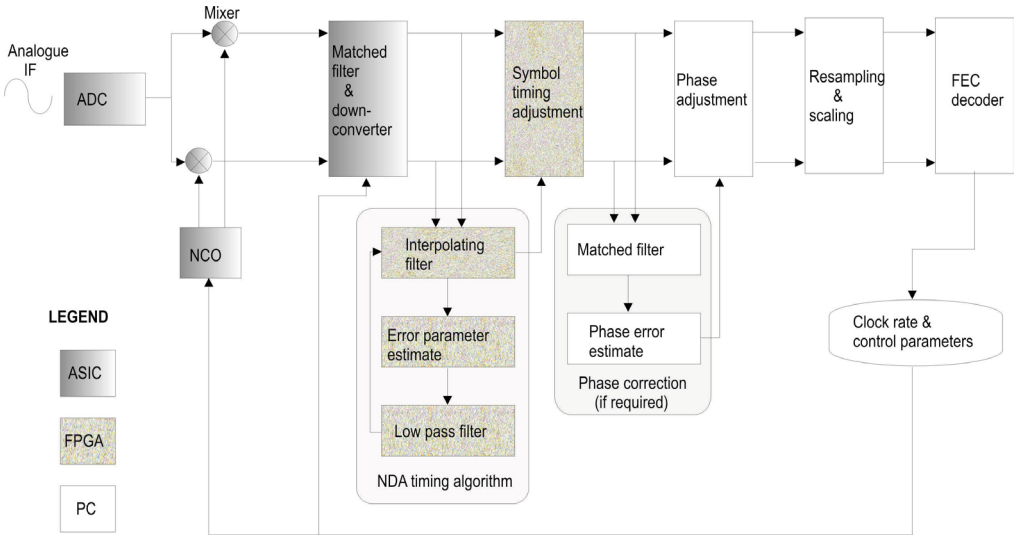

Figure 1 illustrates the signal processing blocks required at the receiver, as well as the types of devices which perform the processing.

With the above requirements in mind, the platform was essentially constructed using four COTS components:

1. Dual Opteron desktop PC
2. Radio card (Pentek)
3. Linux Operating System
4. SCA version 2.2 (CRC)

The prototype is comprised of a standard ATX form factor desktop PC board running dual 2.1 Gigahertz (GHz) Opteron processors. The intermediate-frequency (IF) to base-band conversion is performed on a self-contained receiver board distributed by Pentek Inc., which connects directly to the PC's Peripheral Component Interconnect (PCI) slot. The operating system was the open source Linux Fedora Core 2 distribution using a recompiled 2.6.9 kernel.

The SCA was developed by the CRC in C++ and uses the TAO ORB. The SCA is itself a core framework (SCA-CF) of standards developed under the Joint Tactical Radio System (JTRS), a Defense Advanced Research Project Agency (DARPA) initiative aimed at streamlining the interoperability of tactical United States military radio systems. The development of the SCA compliant DVB-S waveform was conducted by the CRC, while the DVB-S2 waveform and integration of the platform itself was performed by Array Systems Computing Inc.

The functional design of the receiver is shown in Figure 1. The ASIC components on the radio card are responsible for the frequency shifting of the IF input, and then performing the filtering and decimation. Onboard the radio card is a Vertex-II Pro FPGA, which is available for users to insert application specific signal processing code. In this case, a non-data aided (NDA) timing and synchronization algorithm (designed for M-PSK demodulation) was first prototyped in C and then coded in VHDL. Using the Pentek GateFlow (which details the connections and the interfaces necessary for the user to modify and/or insert their own signal processing (SP) code on the FPGA) along with the Xilinx ISE8.2, the demodulator SP code was compiled and loaded into the Virtex-II Pro FPGA.

The phase correction, scaling, and framing required by the DVB-S2 standard were chosen to be performed on the PC. The reason for this was two fold: (1) Depending on the deployed waveform, phase corrections and the subsequent framing of data are performed at different points along the SP chain. For example, for symbols coming out of the demodulator DVB-S requires the symbols have a $\pi/4$ ambiguity, whereas DVB-S2 uses a physical layer (PL) header to allow the demodulator to completely correct the phase of the symbols; (2). The second reason was that this was in line with the goals of keeping the design as generic as possible, and wherever feasible, to perform as much of the SP as possible on the GPP devices.

While not shown here, the diagram of the transmitter is functionally equivalent to the receiver. The PC performs the requisite Forward Error Correction (FEC) coding and framing of the base-band file. The symbols are then mapped using Gray encoding to the target M-PSK symbol set. The data is subsequently sent to the FPGA where it is up-sampled and base-band filtered. An Intellectual Property (IP) core from Xilinx was inserted on the FPGA (again via the GateFlow) to implement a Root Raised Cosine Filter. Finally using ASIC components, the output was sent to a quadrature modulator and mixed with the appropriate IF carrier.

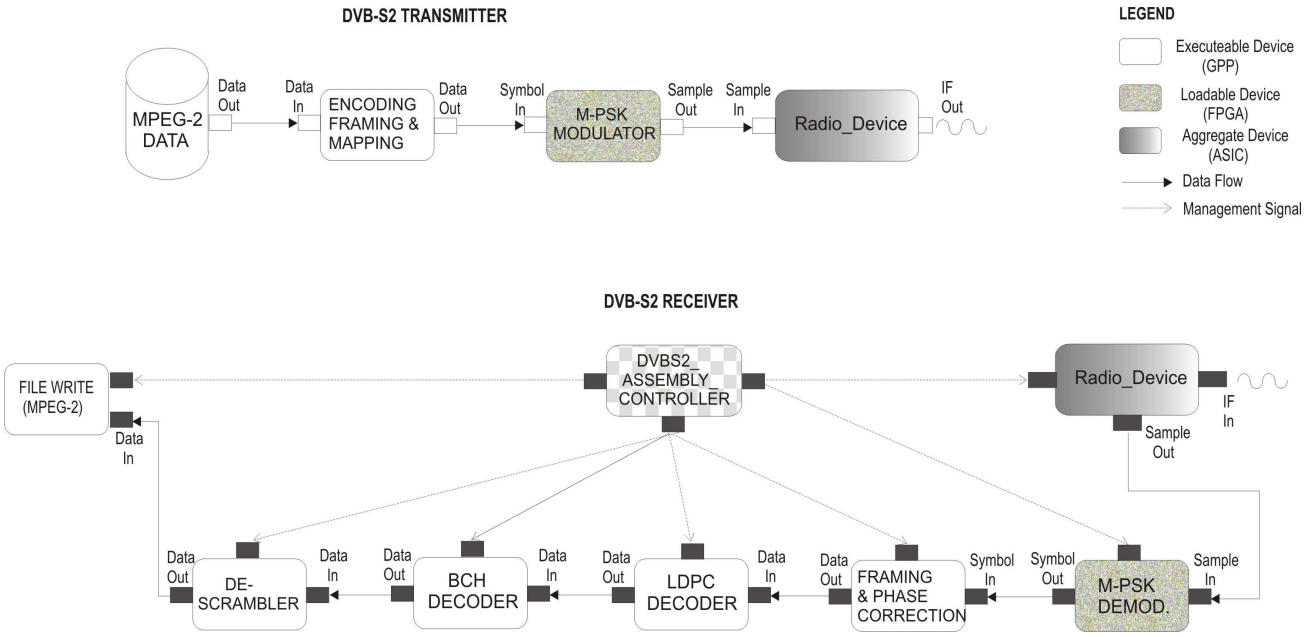

Figure 2 illustrates the data flow of the DVB-S2 receiver application. The coloring of each resource component denotes the type of device that the resource uses.

3. WAVEFORM DEVELOPMENT

To implement the DVB-S2 as an SCA waveform, the goal was to implement each of the functional components (baseband framing, Forward Error Correction Codes (FEC), physical layer framing) outlined in DVB-S2 standard as *resources*. This idea is outlined in Figure 2, which shows the flow control for the receive portion of the SCA DVB-S2 waveform. Additionally, the type of device where the resource is deployed is also illustrated. For example, the resource corresponding to the SP required for Framing and Phase Correction is deployed on a GPP. Note that since the granularity (i.e., how much processing each resource performs) of each resource is left to the designer, and that each resource may be executed on a different, or the same, executable device (e.g. GPP), there is a great deal of flexibility for how to 'develop & deploy' the waveform.

The resources are inter-connected to each other through ports, and are controlled via the *assembly controller*. The actual signal processing is performed in the input port of the resource. This is shown in Figure 3. Data are passed from the output port of one resource to the input port of the next resource in the form of blocks of bytes. For example, when a block of bytes arrives at the input port of the LDPCDecoderResource, the processData(...) function in the input port is invoked to process this incoming data block. In this case, the processData(...) function performs the Low Density Parity Check (LDPC) decoding algorithm on the incoming data block. Once the processing is completed by the algorithm, the processed data block is forwarded to the resources' output port, where a CORBA function is called to:

1. Send the processed data block to the input port of the next resource (i.e. in this case, the BCHDecoderResource),
2. Invoke the processData(...) function in the input port of the next resource to start new processing (i.e. BCH decoding) on the data block.

Thus as a first step to convert the SP components in a wireless standard into SCA resources, the functionalities of the standard are implemented in the processData(...) function of each resource's input data port.

4. VALIDATION AND TESTING

The end objective of this ongoing prototype development is to design and develop a software reconfigurable radio platform capable of implementing the DVB-S waveform as well as limited functionality of the DVB-S2 waveform in a manner suitable for OBRP. To achieve this objective, the project aims are a set of validation experiments designed to show real-time processing, upgradeability, and interoperability. For example, the prototype system will target real-time processing of DVB-S at data rates up to 2 Megabits per second (Mbps) and real-time processing of DVB-S2 at data rates up to 1 Mbps.

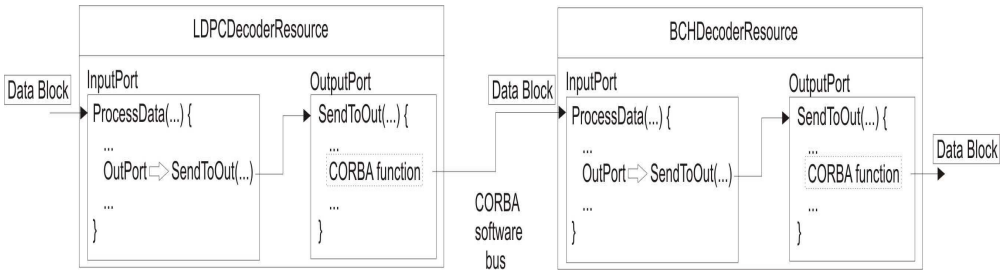

Figure 3 shows one method for 'wrapping' the signal processing code within the SCA, as well as the data flow for how data passes from one resource to the next.

As a first step in the development of the DVB-S2 waveform, the receive portion was first prototyped in C++ and then 'wrapped' using the SCARI++ Software Suite from CRC. The (de)modulator was also simulated using both floating, and fixed point C++ representations. It was then modelled in VHDL, inserted into the GateFlow and compiled using ISE8.2. The bit file was then loaded on the FPGA.

The aim was to first upload multiple applications (of the same or different waveforms) to the radio, and then instantiate one of the DVB-S2 applications. The receiver should then be capable of providing a 1Mbps throughput. Using the Radio Manager it was straight forward to load multiple applications of the DVB-S2 waveform with varying attributes (for example, varying the number of iterations the LDPC codes would run), or even loading a different waveform such as the DVB-S waveform altogether. However, since they compete for the node1PentekDevcie (the radio card), only one application could actually be instantiated at a time. It was estimated that the two bottlenecks in performance would be the matched filtering for estimating the phase error and the iterative process of decoding required by the low density parity check codes (LDPC). Benchmarking of the two individual processes showed that both were capable of exceeding 1Mbps; however, the PC (represented as a single node with one executable device – the GPP) was overwhelmed when simultaneously running all signal processing components of the waveform.

In response, a second node (PC) was added and the waveform was bifurcated so that the resources corresponding to the phase correction and framing would be performed on one node, while the FEC resources would be deployed on a second node. Figure 4 shows this modification as depicted using the Radio Manager in the SCARI Software Suite (the arrows show the connections between the components). The figure indicates three distinct nodes. The radio card is depicted as 'node1PentekDevice' and it sends data to the physical layer descrambler resource which is deployed on the GPP of the PC. Once the data has been phase corrected and properly formatted, it is sent to the decoding resources deployed on the 'node2Executeable Device'. After the necessary processing has been performed on node2, the decoded data are sent to the third node, which is simply a laptop used to display the decoded MPEG-2 clip.

It was found that when the waveform was deployed and started, there were two additional sources which also consumed a large proportion of the GPP's resources. The first was the device driver for the radio card, as it is needed to perform the buffering of the data as it moves from RAM across the PCI bus for both the modulator and demodulator. The second and more pronounced, was the process of transferring data via the 'provides/uses' ports of the resources. It was found that if data was forced at a higher rate than the waveform could process, the data would simply be dropped, or would cause the waveform to fail. In this case, the waveform would have to be shutdown and reinitialized. Current throughput rates on the waveform illustrated in Figure 4 are approximately 500kbps. While this is below the target rate, this work is ongoing; and no concerted attempt to optimize the code or the flow processes have been made. Further investigations towards optimizing the throughput of the waveform on the current hardware will undoubtedly yield superior rates.

5. SUMMARY

The development of this SDR test-bed is a logical first-step towards a target broadband regenerative system in which the (de)modulation, (de)coding, and switching is performed locally, onboard the radio platform itself. Initial results using GPPs to perform most of the requisite signal processing and all the radio management indicate that while achieving large real-time data rates remains a challenge, the system was found to be easily upgradeable and could instantiate multiple waveforms. As this work is ongoing,

further development and experimentation are needed to help realize the benefits of not only using the SCA to help regulate radio management and force standardization, but also in the use of SDR principles in general.

REFERENCES

[1] M. Ibnkahla, A. Sulymna, H. Al-Asady, J. Yuan, and A. Safwat, "High-Speed Satellite Mobile Communications: Technologies and Challenges," *Proc. of the IEEE*, Vol. 92, No. 2, pp. 312-339, Feb. 2004.

[2] http://jtrs.spawar.navy.mil/sca/

[3] Digital Video Broadcasting (DVB), European Standard (Telecommunications series) EN 300 421, V1.1.2, 1997-08.

[4] Digital Video Broadcasting (DVB): Second Generation, European Standard (Telecommunications series) ETSI EN 302 307, V1.1.1, 2004-06.

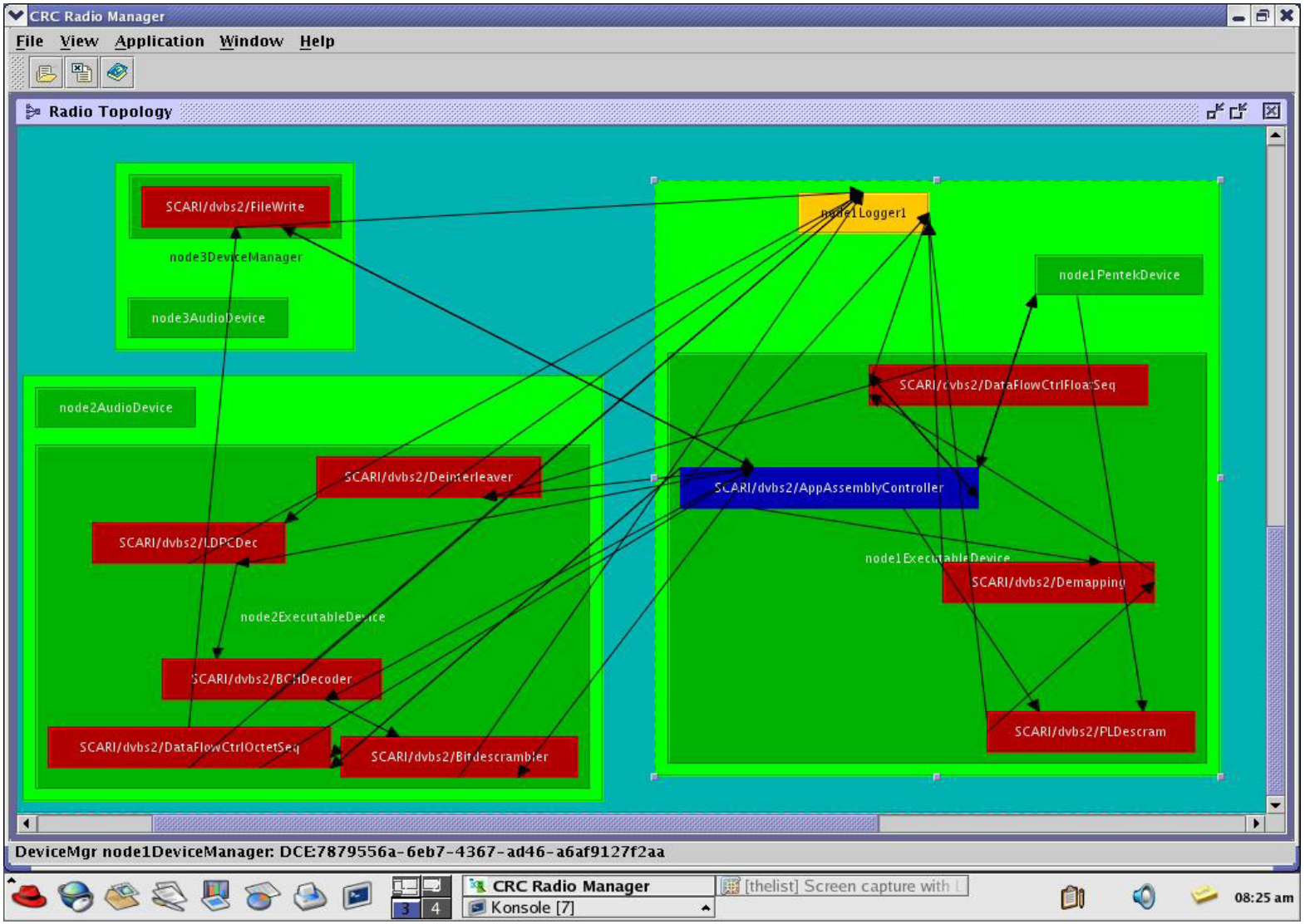

Figure 4 illustrates the user interface using the Radio Manger in the CRC SCARI++ software package. The figure depicts how the DVB-S2 receive waveform is instantiated across three nodes (which correspond to three PCs). Additional information such as data flow, what devices are available, and the location of which resources are deployed on which devices are also shown.