From July 2007 *High Frequency Electronics*

Software Defined Radio Architecture for NASA's Space Communications

By Maximilian C. Scardelletti, Richard C. Reinhart, Monty Andro, NASA's John H. Glenn Research Center; Dale J. Mortensen, Thomas Kacpura, ZIN Technologies; Carl Smith, John Liebetreu, General Dynamics—C4S; and Allen Farrington, Jet Propulsion Laboratory

NASA is developing a software-defined radio standard for future space communications that allows both hardware and software reuse, system scalability, and practical hardware upgrades

The National Aeronautics and Space Administration (NASA) is developing a standardized platform for future space communications needs, based on software-defined radio (SDR) concepts. The goal is to have a system that does not require a complete custom hardware/software design for each mission. Instead, NASA desires an adaptable and scalable system that can accommodate the performance demands of many different radio systems. In addition, the hardware/software architecture is intended to protect the intellectual property rights of contractors regarding the internal operation of the portions of the system they provide.

Introduction

This study defines a hardware architecture approach for software defined radios that will enable commonality among NASA space missions. The architecture accommodates a range of reconfigurable processing technologies including general purpose processors, digital signal processors, field programmable date arrays (FPGAs), and application specific integrated circuits (ASICs) in addition to flexible and tunable radio frequency (RF) front ends to satisfy varying mission requirements.

The hardware architecture consists of modules, radio functions, and interfaces. The modules are a logical division of common radio functions that comprise a typical communication radio. This paper describes the architecture details, module definitions, the typical functions implemented by each module and the module interfaces. The trade-offs between component-based, custom architecture and a functional-based, open architecture are also discussed. Readers should note that the architecture does not specify an internal physical implementation for each module, nor does it mandate the standards or ratings of the hardware used to construct the radios.

Advantages of SDR

A software defined radio is a collection of hardware and software technologies that is reconfigurable to provide a variety of systems for communication networks. Software defined radios are programmable systems with partitioned software and hardware modules controlled by managing software conforming to defined interfaces, to allow design reuse, software portability, and to provide scalability across hardware platforms. The advantages of flexible and adaptable operation in the digital and RF domains offer significant capabilities and performance compared to legacy radios with fixed, dedicated functionality.

Software defined radio has the potential to save missions cost when building multi-mode, multi-band, multi-functional radio systems that can be dynamically changed using software upgrades. In other words, the same piece of "hardware" can be modified to perform different functions at different times through reprogrammable software modules, reducing the total number of radios required for a given mission. Software defined radios based on a standard architecture provides NASA with a consistent and common framework to develop, space qualify, operate and maintain these complex reconfigurable and reprogrammable radio systems for space.

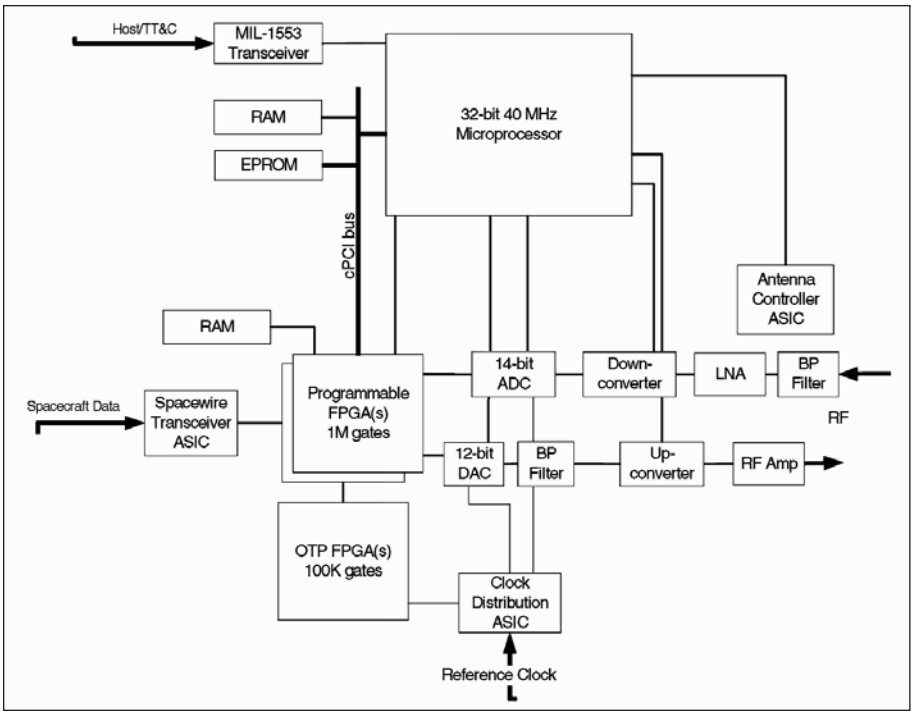

Figure 1 · Custom implementation hardware architecture.

A standard open architecture has published interfaces enabling different vendors to provide radios that meet the interface standard, providing commonality among different vendor implementations, and enabling interoperability between providers of different hardware and software. Open interfaces also allow flexibility for component replacement and technology insertion. Commonality among radios, design reuse, and software reuse potentially reduce NASA mission risk. The Space Telecommunications Radio System (STRS) Architecture [1] is one approach under consideration to establish a standard open architecture for NASA's SDR developments.

As NASA looks to the future of space exploration, the growth of reconfigurable electronics and software defined radio provides an opportunity to change the way space missions develop and operate space transceivers for communications and navigation.

Custom Architecture

A software defined radio architecture is: "a comprehensive, consistent set of functions, components, and design rules according to which radio communication systems may be organized, designed, constructed, deployed, operated and evolved over time. A useful architecture partitions functions and components such that a) functions are assigned to components clearly and b) physical interfaces among components correspond to logical interfaces among functions" [2].

Current implementations for space deployed SDR hardware are typically represented in terms of the device components which comprise the software defined radio. Figure 1 shows a high-level, component-based, custom hardware architecture. The custom approach emphasizes typical hardware transceiver elements such as ASICs, FPGAs, specific memory elements (e.g. RAM, EEPROM), physical interfaces (RS-232, Spacewire, etc.), and RF signal conversion and filtering components (e.g., BPF, specific ADC). These are generally represented using available technology.

The diagram illustrates custom connections between radio processing devices and between the processing elements and the RF front end. These custom radio interface connections are often proprietary to the radio developer.

The custom approach can provide an efficient solution to meet a particular mission's requirements. Design, development, testing, and space qualification are specific and unique for that implementation and radio design. However, modification of the design and new software development is often required to accommodate new parts required due to obsolescence and to allow for technology insertion to meet new mission requirements.

The proprietary interfaces limit NASA's ability to extend and reuse the investments in software developments from one mission to another. Also, the custom implementation is limited to missions with a similar set of requirements and requires NASA to use the same provider. Although hardware is reused, the need for a specific vendor's radio increases costs due to loss of competition.

STRS Open Architecture

Figure 2 shows the current STRS open hardware architecture [3]. The open hardware architecture emphasizes the radio functions and key interfaces. The radio functions are distributed among different modules, to organize different platform services and waveform functions within the radio. Modules are a logical division of functionality to maintain common interface descriptions, terminology and documentation among SDR developments. A waveform comprises the end to end functionality (e.g. modulation, coding, frequency conversion, filtering) and bidirectional transformations applied to information content that is transmitted over the air.

The three major modules of the architecture are shown, illustrating the command and control, signal processing, and analog portion of a radio:

The General Purpose Module

(GPM) provides the basic software execution processes based on general purpose processors. The GPM is a required module supporting the operating environment responsible for waveform instantiation and execution, radio services, and hardware abstraction. The Signal Processing Module (SPM) and RF Module (RFM) conduct signal processing and RF front end functions, respectively. Other modules not explicitly shown include security, networking and optical as required by the transceiver. The radio developer has the flexibility to combine these modules and their functionality as necessary during the radio design process to meet the specific mission requirements.

Several key external interfaces of the architecture include Host Telemetry, Tracking and Command (TT&C), Ground Test, Data, Clock, and Antenna. The Host/TT&C interface represents the low-latency, low-rate interface for the spacecraft (or other host) to communicate with the radio. This type of information includes health, status, and performance parameters of the radio and the link in use. In addition, this telemetry often includes radiometric tracking and navigation data. Information delivered through this interface includes configuration parameters, configuration data files, new software data files, and operational commands.

The Ground Test Interface is exclusively used for ground-based integration and testing functions. It typically provides low level access to internal parameters that are unavailable to the Spacecraft TT&C Interface.

The Data Interface is the primary interface for information that is transmitted or received by the radio. This interface is separate from the TT&C interface because it typically has a different set of transfer parameters (protocol, speeds, volumes, etc) than the TT&C information. This interface is also characterized by medium to high latency and high data rates.

The Clock Interface is used for receiving the frequency reference required to support navigation and tracking. This type of input frequency reference is essential to the operation of the radio and provides references to the SPM and RFM.

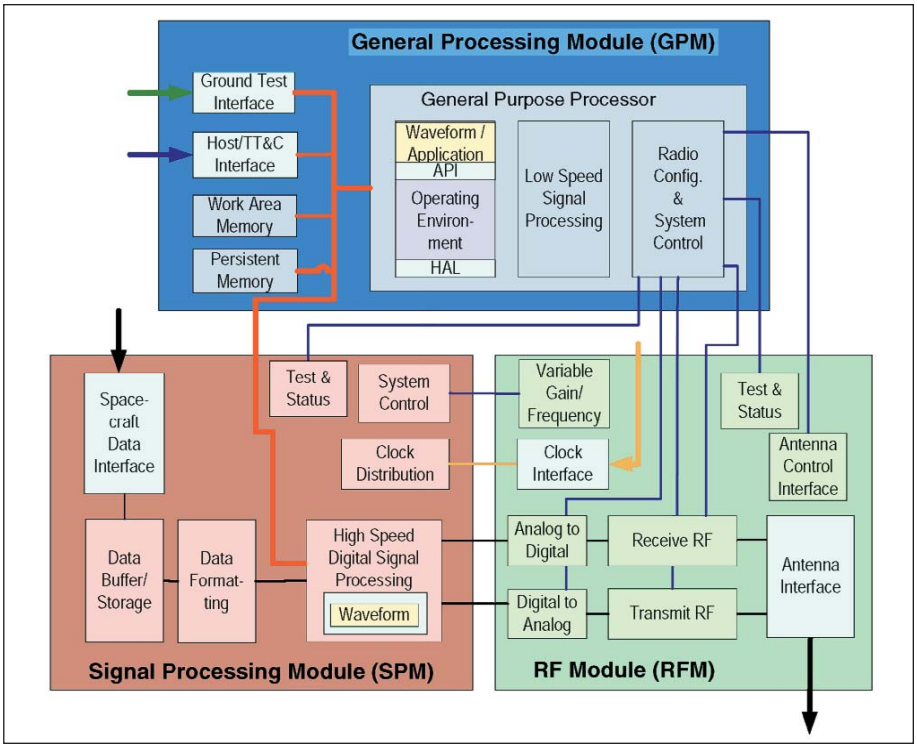

Figure 2 · Open hardware architecture.

The Antenna Interface is used for connecting the electromagnetic signal (input or output) to the radiating element or elements of the spacecraft. It also includes the necessary capability for switching among the elements as required.

Some internal interfaces must be defined in an open manner to support the overall goals of the architecture. Internal interfaces include the system bus between the GPM and SPM, various control lines between the GPM and RFM, ground test interface to each module from the GPM, and frequency reference from the RFM to the other modules.

The System Bus provides the primary interconnect between the GPM's microprocessor and memory elements, interconnection to the external interfaces, and Telemetry, Tracking, and Command and Ground Test Interfaces. The System Bus is the primary interface between the GPM and the SPM. The System Bus provides the interface to reprogram and re-configure elements of the SDR. It supports the read/write access to the SPM elements, as well as reloading of configuration files to the FPGAs.

The interface between the GPM and the RFM is primarily a control/status interface. It is important to have a hardware-based confirmation and limit-check on the software controlling any RFM elements. The system control element of the GPM provides this functionality thus keeping the GPM RFM control bus within operational limits.

The internal Test and Status Interfaces provide specific control and status signals from different modules or functions to the external Ground Test Interface. These interfaces are used during development and testing to validate the operation of the various radio functions.

Finally, the data paths are the

Hardware Architecture Tradeoff Summary

Custom	Open
Cost/power efficient for specific mission requirements	Published interfaces
Applicable to today's technology	Reduces impact of parts obsolescence
Unique design, test, space qualification	Functions abstracted from hardware
Proprietary to specific developer	Retains proprietary radio aspects

Table 1 · The primary tradeoffs between custom and open hardware architectures.

various streams of bits, symbols, and RF waves connecting the major blocks of the primary data-path. For any particular implementation, the data path or bitstreams are defined by the particular waveform implemented in the functional blocks. The interface between the RFM and SPM, however, should be well-defined and should have characteristics suitable for that level of conversion between the analog and digital domains.

This open architecture abstracts functionality away from specific hardware devices through the hardware and software interfaces, enabling greater reuse of a design and minimizing the impact associated with parts obsolescence. Since the software is abstracted from the hardware, there is a greater likelihood of reusing the software in future developments.

The open architecture approach allows NASA to reuse its investment in software radio developments, yet:

1. Maintain company proprietary approaches and designs behind the common interfaces,
2. Reuse the architecture specification among different developments and different vendors, and
3. Preserve commonality among designs, development, testing, and space qualification processes.

Table 1 summarizes the trade-offs between a custom architecture and an open architecture SDR approach for NASA. While the custom approach efficiently meets mission requirements, the approach is limited to today's technologies. The open architecture provides more flexibility to NASA, yet maintains proprietary implementations by the respective developers.

To achieve the benefit of hardware and software reuse, a STRS Repository is envisioned where appropriate transceiver interfaces, documentation and software artifacts submitted by developers are reused by subsequent developers.

Hardware modules that can be used again for other missions reduce cost, system integration, and risk. Using

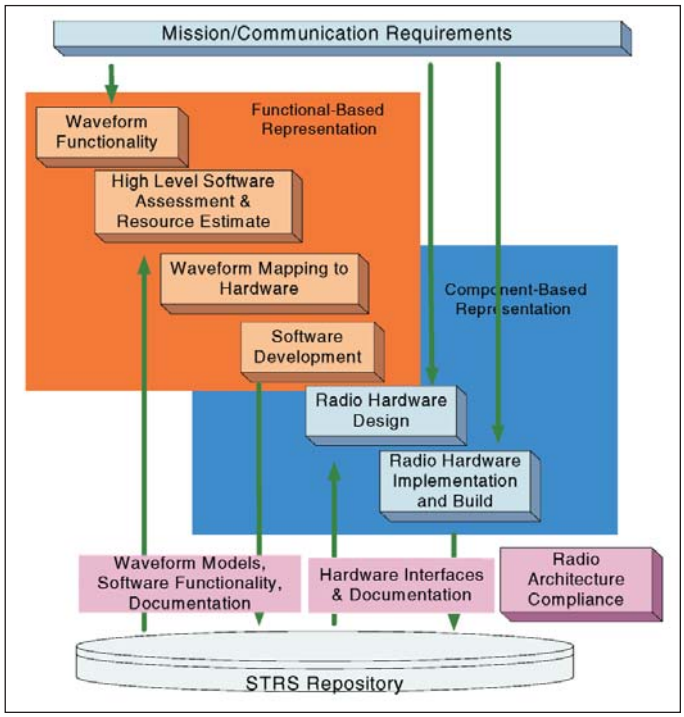

Figure 3 · Radio development process.

previously developed hardware reduces risk and cost since the hardware has space flight heritage, has demonstrated reliability and its space qualification procedures are known.

Most often, the motivation to change hardware is either new performance requirements or parts obsolescence. Defining standard interfaces between modules allows developers to insert new hardware while still reusing other aspects of the radio. The ability to insert new hardware allows multiple vendors to contribute. Software interfaces help mitigate parts obsolescence by reusing software with the new parts, thus saving development time and cost.

Radio Development Process

During the radio development process, one can apply the open hardware architecture from both a function-based view and a more component-based view as the process evolves, as illustrated in Figure 3. In both instances, the architecture benefits emerge where software and hardware modules, documentation, and interfaces can be common among successive developments.

The radio development process begins with an assessment of mission requirements applicable to the communications system (e.g. communications and navigation radio). During this requirements phase the team determines radio and then, ultimately, waveform functionality (e.g. modulation type, coding, filtering, frequency conversion) required for the mission. The hardware architecture

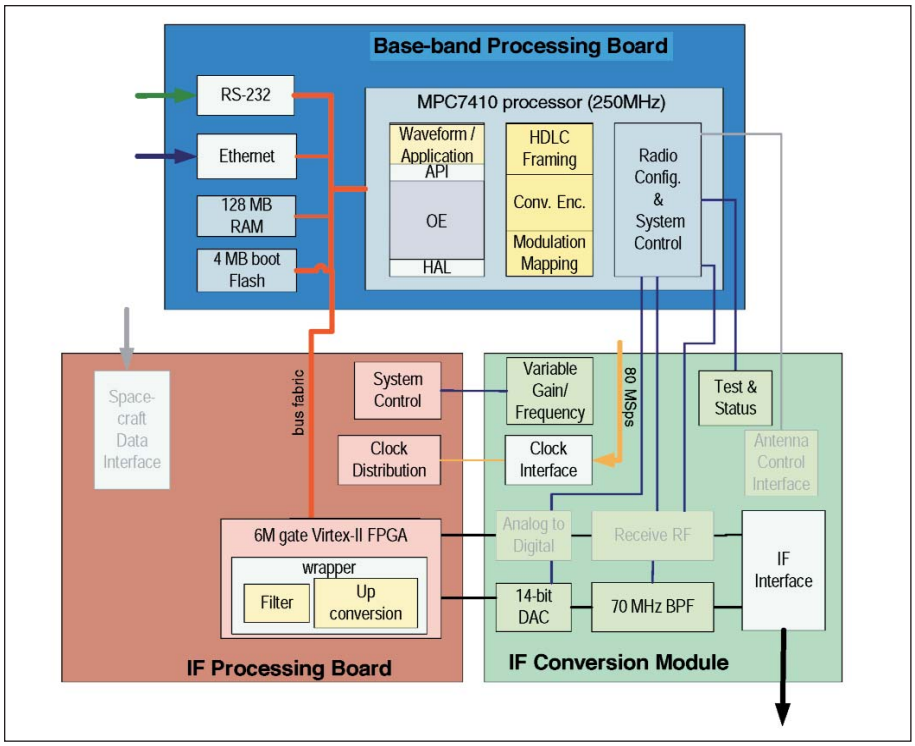

Figure 4 · Transmit waveform applied to hardware architecture.

depicted in Figure 2 illustrates how this functionality can be divided among the various standard modules for consistent terminology and use of common interfaces regardless of developer. As the process illustrated in Figure 3 shows, the mission designers may access the STRS Repository to reuse functionality based in software to reduce time and cost during the design and development phase.

As the transition from functional description to hardware design and implementation begins, the module representation along with published interfaces aids in reuse, test and verification. At this stage, designers conduct a mapping of waveform functions to specific signal processing and RF devices. The common architecture provides waveform independence from the platform developer through the standard interfaces. The interfaces defined by the standard provide an open and published interface, yet protects the intellectual property of the different developers. Using this approach, NASA (or in many cases the prime contractor) could integrate the best hardware modules from different developers into a single product based on the common interfaces.

Applying the Hardware Architecture

Figure 4 illustrates a working example of deploying a reconfigurable transmit waveform on an SDR platform while adhering to the hardware architecture. The example waveform represents typical low rate command/telemetry waveform; QPSK modulation, 1/2 rate coded waveform. Other functions include an internal data generator, HDLC framing, bandpass filtering, and digital upconversion. The waveform functions are deployed across the General Purpose Processor Module, the Signal Processing Module, and the RF Module.

The low rate application data enters the radio through Ethernet interface of the GPM. The data interface of the SPM is not used. The radio is controlled through an RS-232 serial interface for control, and reconfiguration. The GPM handles the low rate functions of the waveform such as HDLC framing, convolutional encoding, and modulation. The data is sent from the GPM to the SPM for filtering, and digital upconversion. The RFM handles the digital to analog conversion, bandpass filtering, signal conditioning (e.g. amplification and IF output. In this case the highest output frequency is 70 MHz, thus a second upconverter is necessary on the RFM to reach a higher RF.

The operating environment (OE) abstracts the low speed signal processing waveform functions from the underlying processor according to the rules of the architecture. In the example several functions of the transmit telemetry waveform were deployed to the general purpose processor module to exercise more functionality of the architecture abstraction. To comply with the STRS Architecture, the developers must provide the radio services described in the STRS Standard and publish the FPGA wrapper interface used in the example. This allows the developer to maintain the proprietary character of their intellectual property associated with the algorithm portion of the filtering and upconversion. This approach exposes the interfaces used in the FPGA for subsequent developers, but protects the investment made by the original developer. The developer must also provide a description of the physical hardware interfaces used in the implementation and a mapping of the control interfaces to each of the modules.

Summary

NASA is considering a standard for software defined radios as they begin to make their way into NASA missions. Commonality among different developments could include software and hardware interfaces, test points, command protocols, models, and documentation. There is a role for

both functional-based, open architecture and device-based representation in the radio development process. Representing the architecture as functions serves as an aid in early mission and radio functionality definition and development. The device oriented representation better applies during radio design and development.

Developers have agreed that standardization applied to software radio hardware will aid development, testing, and verification processes. However, discussions continue on the level of standardization and exactly which interfaces to apply the standard. The development and advancement of the STRS architecture continues to be a multi agency and standards body effort.

Acknowledgements

The authors wish to acknowledge the contributions of the SDR Forum Space Applications Work Group member companies towards this hardware architecture definition.

References

1. NASA, "STRS Architecture Description, Release 1.0," Cleveland, Ohio, April 2006.

2. J. Mitola, *Software Radio Architecture*, John Wiley & Sons, New York, 2000.

3. NASA, "STRS Architecture Standard, Release 1.0," Cleveland, Ohio, April 2006.

4. JTRS, "Software Communications Architecture v2.2.2", Washington D.C. April 2006.

5. R. Reinhart, et al, "Hardware Architecture Study for NASA's Space Software Defined Radio," WAMICON 2006, Clearwater Beach, FL, December 2006.

Author Information

Maximilian C. Scardelletti is a senior research engineer in the Communications Technology Division at NASA Glenn Research Center in Cleveland, Ohio. He received his BSEE and MSEE from the University of South Florida in 1997 and 1999, respectively. He is currently enrolled at Case Western Reserve University, where he is working towards his Doctoral degree in microwave micro-electromechanical systems (MEMS). He can be reached by e-mail at maximilian.c.scardelletti @nasa.gov

Richard C. Reinhart received a BSEE from The University of Toledo in 1989 and a MSEE from Cleveland State University in 1993. He has worked in space communications since 1989, including the Advanced Communications Technology Satellite, Ka-band phased array development, high rate system analysis, and software defined radio. Richard is founder and Chair of the SDR Forum Space Work Group.