Achieving SCA Compliance for COTS Software Defined Radio

Second Edition

Overview
Technology
SDR Products

by
Robert Sgandurra
Product Manager, Pentek, Inc.
The Software Communications Architecture (SCA) mandated by the Joint Tactical Radio System (JTRS) program office provides a software framework for the implementation of Software Defined Radio (SDR) platforms.

This framework provides many features to increase the portability of waveforms by including a common operating environment and a set of common services and standardized component interfaces.

**SCA-Compliant Platforms**

Developers of JTRS platforms must conform to the SCA specifications and undergo certification to ensure that requirements have been met. As a result, JTRS radio set developers who require FPGA or DSP devices to achieve the required levels of performance for complex waveforms and multichannel operation, are faced with two choices: either develop custom hardware and build SCA compliance into it, or buy COTS products and adapt the standard drivers and libraries to SCA compliance.

**Pentek Board-Level Products for SCA**

COTS vendors such as Pentek can provide SCA-compliant hardware products that are fully compatible with a core framework environment and development tools.

The Pentek Model 7640 Dual Channel Software Radio Transceiver is a PCI board that was developed to address SCA compliance. Together with a full-featured C++ implementation of the SCA core framework provided by the Communications Research Center (CRC) of Canada, and a PC running Linux, it allows one to create an SCA-compliant, cost-effective development system.

As with all Pentek board-level products, also available is the Pentek ReadyFlow® Board Support Package (BSP). It provides drivers and development tools for quick board start-up through application development.

FPGA support is provided with Pentek’s GateFlow® FPGA Resources. These include the FPGA Design Kit for custom algorithm development; the FPGA IP Core Library, a collection of highly optimized cores; and the factory-installed IP cores in Pentek board-level products.
Software Defined Radios

In a nutshell, JTRS defines a system for software programmable radios, for reliable multichannel voice, data, imagery, and video communications. It supports platforms from battery operated handheld units to base systems found in headquarters, and everything in between.

The system is modular and scalable to allow more bandwidth and channels as needed. It needs to be backwards compatible with legacy radios that it will eventually replace. It supports transparent, dynamic, intra- and inter-networking routing.

It eliminates stovepipe legacy systems where each component is so intimately tied to the next that trying to replace an outdated component could cause a change to ripple through the whole system. It carries real-time information through, what the army likes to call, the last tactical mile to the fighter in the field. And, in general, it ensures greater success and safety for war fighters through multinational, cross-echelon communications.

JTRS goals go beyond the battlefield. Some of the potential is in applications like US Homeland Security, Federal, State, and Municipal law enforcement, ambulance and emergency medical technicians, fire fighters, search and rescue, land and sea, commercial and private aviation, and international commercial applications.

The JTRS mandates the use of SCA, which is the Software Communications Architecture that allows multiple radio types, or waveforms on one hardware platform, and waveforms that are portable across all platforms. These are two major requirements of the JTRS initiative.

Some of the major features and requirements of JTRS are listed on the JTRS Website. For the URL, see Useful Links on the last page of this handbook.

SCA Core Framework

The figure below shows a very simplified version of the main layers of a JTRS radio set. With it, you can start to get a feel of where all the components come from.

Starting from the bottom and working up, the three major components are:

- The actual radio hardware that provides all the analog and digital interfaces to the outside world.
- The operating system and middleware.
- The SCA core framework which is the software that interfaces to the hardware.
- The waveforms that are actually what gives the radio its specific characteristics needed to satisfy the application requirement.

The SCA core framework connects the SCA-compliant waveforms shown at the top with the SCA-compliant hardware at the bottom within the JTRS radio set. This allows the platform developers who produced an SCA-compliant radio set and the waveform developers, who independently developed their SCA-compliant waveforms, to expect that the waveforms will function properly with the radio hardware.
SCA Operating Environment

Shown in Figure 1 below, is the SCA Operating Environment, sometimes called the OE. It basically depicts all the layers of the software and how they interact.

Figure 2 depicts an expansion of the OE components. Starting with the components below the application line, we will work our way down so we can build a foundation for the applications to run.

These components consist of the Core Framework, the Operating System, the Board Support Package, CORBA (Common Object Request Broker Architecture) ORB (Object Request Broker), and the Network Services. The SCA recognizes the need for separating the black, or secure components of an application from the red, or non-secure components.

So what we have here are basically two parallel and complete OE sets that exist in the same application to support both black (secure) and red (non-secure) components of the system.
Core Framework

The SCA specification defines the Core Framework (CF) as the essential core set of open application layer interfaces and services. The framework provides an abstraction of the underlying software and hardware layers for software application designers.

The core framework includes the base application interfaces which can be used by all the software applications and the framework control interfaces that provide system control. These include:

- The Domain Manager
- The Device Managers, which will be described later
- The Framework Service that supports core and non-core applications
- The Domain Profile that describes the properties of the hardware devices and the software components of the system and domain profiles

These are written in XML (eXtensible Markup Language) which is actually a protocol used for web design and a number of different software development applications.
CORBA ORB Middleware

As a message passing technique, CORBA is used for distributive processing throughout the software environment. Distributive processing is a fundamental aspect of the SCA specification and CORBA is a widely used middleware for cross-platform frameworks. These cross-platforms can be used to standardize client and server operations when using distributive processing.

All core framework interfaces are defined in IDL (Interface Definition Language) which is independent of programming language and can be compiled into programs created in C++, ADA, or Java. The SCA IDL defines operations and attributes that serve as a contract between the components.

Operating System

The Operating System layer is defined by SCA to include real-time embedded operating system functions that provide multithreaded support for applications including the Core Framework applications.

The architecture requires a standard operating system interface for operating system services in order to facilitate portability of applications. The operating system must be POSIX compliant, and POSIX, if you're not familiar with it, is a Portable Operating System Interface, which is an established industry standard. The SCA uses a small subset of the POSIX definition, specifically the real-time controller system profile.
Network Services

The SCA specification calls for Network Services that allow commercial components to support multiple unique serial and network interfaces.

As a result, possible serial and network physical interfaces include RS-232, RS-422, RS-423, RS-458, Ethernet and 802.11 wireless protocols. And, to support these interfaces, low-level network protocols may be used. These include PPP, SLIP, and others.

Elements of waveform networking functionality may also exist at the operating system level. An example of this would be a commercial IP stack that performs routing between waveforms.

Board Support Package

The Board Support Package, basically the last layer, is the closest to the hardware functionality of the actual hardware platform.

The SCA specification allows for operation on commercial bus architectures, so the operating environment can support reliable transport mechanisms. These may include error checking and correction at the bus support level.

Possible standards include VME, PCI, CompactPCI, Firewire, Ethernet and the new high-speed serial interfaces.

Again, the SCA operating environment recognizes the use and the need of red (non-secure) and black (secure) components of the system, so you can see that there are two different sets of Board Support Packages, one for each side of the application.
SCA Applications

If we now move to the Applications group, we’re looking at applications which include functions such as modem level DSP, link and network protocols, I/O access, security services, and others.

As you’d expect, the applications must use the Core Framework Interfaces and Services. The applications consist of one or more Resources and the resource interface provides a common API (Applications Programming Interface) for the control and configuration of software components.

The various hardware elements of an SCA-compliant radio are referred to as Devices, and they can include GPPs, which are the General Purpose Processors, A/Ds, D/As, Digital Receivers, Upconverters, FPGAs, DSPs, and other types of equipment.

From a software point of view, the software proxies that act to interface to the hardware are also referred to as Devices. This may be a little confusing but in the overall theme it does work, since all communications to the underlying hardware must flow through these structures.

Devices

Devices are divided into four classes. Simple elements like A/D and D/A converters that require basic control monitoring and data connections are just simply called Devices.

More complex elements like GPPs capable of accepting and executing program code are called ExecutableDevices.

An FPGA is capable of performing a signal processing function, but typically it cannot really execute program code, although we will show later how embedded PowerPC cores in the FPGAs can do some execution. In general FPGAs must first be configured for a particular task, so they’re called LoadableDevices.

The fourth class of a Device is a combination of Devices, so it’s termed the AggregateDevice. An example of this class would be a board that contains both A/Ds and FPGAs.
Adapters

The last key components defined by the SCA are the Adapters. Adapters are Resources or Devices used to support non-CORBA compatible elements in the domain. Therefore, adapters are used to provide that translation between the non-CORBA compatible component and the CORBA compatible resources.

Since an adapter implements the core framework CORBA interface, known to other CORBA compatible resources, the translation service should be transparent to the CORBA compatible resources.

Some examples of Adapters are modems, security I/O, and host processing elements.

Next Topic

This concludes our overview of the general architecture of an SCA system and the operating environment.

The next topic we will cover and build upon is the use of FPGA technology in Software Defined Radios.

It’s important to understand that a key component to fully integrate the FPGAs into an SCA system is still missing. However, FPGA IP (Intellectual Property) cores can be integrated into an SDR solution.

SCA is working right now to catch up to the technology and best incorporate the features of FPGAs into the specification.
Achieving SCA Compliance for COTS Software Defined Radio

FPGA Roles in an SDR Product

FPGAs are a crucial part of many SDR systems due to the outstanding performance they provide. While dynamic loading and control of FPGAs is still not a mature part of the SCA specification, FPGAs can be preconfigured to extend the hardware performance and functions of a system component outside the SCA specification. Some of their key advantages include:

- **Replacement for Interfacing and Glue Logic.** The FPGA flexible I/O pins can mimic various logic levels to implement a wide range of high-performance interfaces to components, buses, memory devices and communication ports. In the hardware example shown, note that all the resources are connected directly to the FPGA which is the center point of the whole product. Not only does the FPGA provide the necessary logic interfaces, it can also provide flexible and reconfigurable interconnects between the resources.

- **Control Structures.** The FPGA includes specialized hardware structures including synchronous DRAM controllers for the three target memories. An advanced dual clocking, timing, and synchronization system built into the FPGA logic simplifies data acquisition and time stamping, and supports multichannel operation.

- **Custom Signal Processing.** Another role of the FPGA is to provide custom signal processing afforded by its DSP structures such as block RAM, hardware multipliers and logic slices. Users can add their own IP or install commercial off-the-shelf IP for specialized tasks. This often saves months of development time and offloads signal processing chores from other processors in the system. In this role, this newly-acquired function of the FPGA is often fixed during deployed operation.

- **Reconfigurability.** In this role, the FPGA is often reconfigured during system initialization or even during runtime, based on the current needs of the application. Here is where FPGAs will further extend the scope of software defined radios.

- **PowerPC Processors.** This role involves exploitation of the on-chip PowerPC processors that can be equipped with an operating system and programmed to execute C code. Since the processors are surrounded by a wealth of device interfaces, memory, and real-time signal processing hardware, they may not be allocated as freely as a generic processor within a pool of general purpose processors. However, these immediate resources may be extremely efficient for data flow and for real-time processing tasks.
FPGA Technology for SDR

New Device Technology

FPGA technology is advancing constantly and we will just highlight a few of the hottest FPGA functions:

❖ 600 MHz DSP slices including multiplier and accumulator
❖ Up to 512 dedicated on-chip hardware MACs (multiplier/accumulators)
❖ Up to 200,000 logic cells
❖ On-chip PowerPC RISC microcontroller cores
❖ Gigabit Ethernet media access controllers
❖ Reduced power with core voltages near 1 volt
❖ Memory densities of over 10 million bits coupled with very flexible memory structure to meet a wide range of dataflow strategies
❖ 500 MHz flexible memory structures for RAM and FIFOs
❖ Logic densities of over 10M gates
❖ Silicon geometries to 90 nanometers
❖ High-density BGA and flip-chip packaging
❖ On-board 11 GHz switched fabric serial interfaces
❖ Over 1200 user I/O pins
❖ Configurable logic and I/O channel interface standards

Xilinx Virtex FPGA Generations

Just to give you a view of some of the features of the FPGAs from the two main providers, Xilinx and Altera, we will focus on the Xilinx Virtex product line and the various devices that represent their high performance line over the past few years.

FPGAs continue to evolve to meet the needs of software, radio, and DSP in general applications. Specifically, the multipliers are essential elements for DSP applications. Multipliers were introduced in the FPGAs with the Virtex-II family, and continue to be a key feature of the latest Virtex-4 products.

There are up to 512 multipliers available in the SX55 part of the new series which directly targets DSP applications. What we're seeing more and more is that as new waveforms are imposing unrealistic processing loads for general-purpose processors, these loads can best be handled with the FPGAs.

With the Virtex-II Pro family, multigigabit serial I/O interfaces were introduced, and these facilitate distributed processing with switch fabrics, or even just a very fast point-to-point connection for data streaming between resources.

These serial I/O interfaces support protocols like PCI Express and Serial RapidIO. In the latter case, FPGA implementations of the protocols have been the only ones available prior to the recent introduction of processors and switches with built-in Serial RapidIO interfaces.

<table>
<thead>
<tr>
<th></th>
<th>Virtex-E</th>
<th>Virtex-II</th>
<th>Virtex-II Pro</th>
<th>Virtex-4 SX</th>
<th>Virtex-4 FX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic Cells</td>
<td>XCV600E</td>
<td>XC2V3000</td>
<td>XCV2VP50</td>
<td>XCE4VX55</td>
<td>XCE4VF100</td>
</tr>
<tr>
<td></td>
<td>15,552</td>
<td>32,256</td>
<td>53,136</td>
<td>55,296</td>
<td>94,896</td>
</tr>
<tr>
<td>Max Block RAM (bits)</td>
<td>295k</td>
<td>1,728k</td>
<td>4,176k</td>
<td>5,760k</td>
<td>6,768k</td>
</tr>
<tr>
<td>Max I/O User Pins</td>
<td>512</td>
<td>720</td>
<td>952</td>
<td>640</td>
<td>768</td>
</tr>
<tr>
<td>18 x 18 Multipliers</td>
<td>-</td>
<td>96</td>
<td>232</td>
<td>512</td>
<td>160</td>
</tr>
<tr>
<td>405 Power PC Cores</td>
<td>-</td>
<td>-</td>
<td>2</td>
<td>-</td>
<td>2</td>
</tr>
<tr>
<td>Rocket I/O Serial</td>
<td>-</td>
<td>-</td>
<td>18</td>
<td>-</td>
<td>20</td>
</tr>
<tr>
<td>Gbit ENET Ports</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>4</td>
</tr>
</tbody>
</table>

Xilinx FPGA Family
FPGA Technology for SDR

FPGA IP Cores

There are a lot of different IPs (Intellectual Properties) available for FPGAs. Even though hardware PowerPC processor cores are now being incorporated right into the silicon of the Xilinx Virtex Pros and Virtex-4 families, user-installed software IPs are available for dozens of popular microprocessors and controllers, even some legacy processors like Z80s.

Wireless and telecom cores include many of the latest modulation and encoding schemes, like Bluetooth and wideband CDMA schemes.

Connectivity solutions which are critical for a lot of the high-performance applications, include many standard system buses like PCI and PCIX, new switch fabric technologies like RapidIO, Serial RapidIO, and HyperTransport.

The IPs embrace both the physical layer interfaces and some of the protocol engines and serial interfaces which reach speeds up to ten gigabits per second.

Sources of FPGA IP Cores

So where does all this IP come from? It comes from a number of sources:

- FPGA manufacturers themselves are typically one source. They have libraries or reference designs and these are usually free to download, but usually have limited support.
- The next step up in terms of performance and support are the cores that are licensed, and these are typically downloaded for free evaluation following which, they can be purchased. The license that comes with the purchase may be a limited license, or may be a production license for deployment with high volume.
- Consultants are also a good source of IP, especially for custom development where the spec is well defined. They can come in and basically implement the specification.
- Third party vendors also offer off-the-shelf solutions. Since a lot of these vendors specialize in specific application areas, many of these cores are really optimized for those applications, so you can get a standard core that's very specific to your application with the performance you need.
**Pentek GateFlow® FPGA Resources**

GateFlow is Pentek’s flagship collection of FPGA Design Resources. It allows extending the performance and features of FPGA based hardware, independent of the SCA. The GateFlow line is compatible with the Xilinx Virtex products and is available as three separate offerings:

- **The GateFlow FPGA Design Kit** includes design information and software files to deploy unused FPGA resources for implementing user-defined algorithms.

  As shown in the diagram below, the yellow box represents the resources in the entire FPGA. The hardware components connected to the FPGA require some form of logic within the FPGA to support hardware specific functions. These interface blocks are represented in green.

  The rest of the FPGA shown in pink, is available for user-defined functions. The GateFlow FPGA Design Kit provides the developer with project files and documentation that can be used in conjunction with the Xilinx development tools to access the various hardware peripherals and data flows.

- **The GateFlow IP (Intellectual Property) Core Library** is a collection of highly-optimized Pentek IP Cores, fully compatible with the GateFlow FPGA Design Kit.

  These cores include general-purpose functions such as:
  - High-speed pipelined 1024- and 4096-point FFTs
  - High-speed narrowband downconverters
  - High-speed wideband downconverters
  - Multichannel narrowband downconverters
  - Pulse compression algorithms

- **The GateFlow Installed Cores** are high-performance, off-the-shelf signal processing functions installed at the factory as firmware in the FPGAs of Pentek boards.

  For end users who have no interest in developing FPGA IP, or maybe don't have the engineers available to do it, we offer many of the IP cores as an installed option on some of our hardware products. Basically the IP function becomes a part of the overall board feature set, and the radio developer accesses the function through our standard board BSPs and drivers.
COTS Hardware for SCA

Build vs. Buy Tradeoffs

We will now turn our attention to the buy versus build decision, when it comes to COTS hardware in SCA systems. The usual factors of this decision still apply, but we have to consider an additional set of variables when we look at the various software layers and tools needed for SCA systems.

So what are some of those advantages and disadvantages? These are some of the tradeoffs that you’re looking at when you’re either building or buying a hardware component for such a system.

Build Advantages

 kursi
The biggest advantage is producing an exact hardware match for the requirements of your system.

 kursi
Similarly, if we know the software requirements at the time of hardware design, the hardware has a better chance of matching the software requirements and meeting an existing software spec or API.

Buy Advantages

 kursi
The product is probably available off the shelf, or, if it’s newly introduced, it’s going to be available soon.

 kursi
COTS vendors are competitive, so there’s always pressure on them to keep their sales prices low.

 kursi
In all likelihood, you have the option to buy similar products from an alternate source—if there’s a problem with the prime vendor.

 kursi
Very often a vendor’s product line will include multiple versions of a product introduced over a number of years with the latest technology in each version.

 kursi
The latest software technology should be available with the minimum effort if the APIs, the BSPs and the drivers were written with a long-term goal in mind.

Build Disadvantages

 kursi
On the down side, if you’re not a hardware design and manufacturing business with the appropriate engineering resources in place, you can be affected by delays in availability of the hardware.

 kursi
You also will run into development costs which may or may not be covered throughout the product life.

 kursi
Development risks can come in many forms and, if you built a product in house, you may not have an alternative source in case there is a problem.

 kursi
The technology may expire and component obsolescence is always a problem.

 kursi
The product can continue to require ongoing engineering for years to come.

Buy Disadvantages

 kursi
The hardware is most likely not an exact match for the system specifications. It will often match the application pretty well, but in some cases there may be no real perfect fit with a COTS product. In this case, we may have to reduce or change the system spec to match the hardware available.

 kursi
Software could also need some work, especially if there’s an existing software application that needs to be ported to support the new hardware. This is where quality and complete BSPs and drivers from the vendor can really minimize supporting efforts.
What requirements does a COTS vendor need to meet in an SCA-compliant software radio board design, so that the product is flexible enough to satisfy custom, in-house development requirements?

**A/Ds and D/As: Devices**

What’s needed here is the flexibility of software-configurable clocking, gating, triggering modes, and any other mode of the A/D or D/A. The data paths need to be configurable and high bandwidth data paths must be included to satisfy typical frequency requirements of software radio systems.

**Other ASICs: Devices**

Digital downconverters and upconverters, would also be considered SCA devices. They, too, need to be fully programmable with the capability to be bypassed for flexibility, should the need arise.

As an example, consider a TI chip downconverter on a board that has a programmable decimation range that’s consistent with most applications. However there are a few applications where the decimation isn’t low enough. In this case, the downconversion, with the proper decimation, may be performed in an adjacent FPGA using a downconverter IP core.

And while something like the ASIC downconverter can be an extremely useful asset to the overall processing, it’s not really consistent with the spirit of the SCA spec, which wants flexibility so it can address all applications. Again, having that chip as a bypassable resource is a way to use it when you need it, or have it out of the way, if it’s not needed for the job.

**FPGA Resources: LoadableDevices**

The next consideration are FPGA resources, which are classified as *LoadableDevices*. As we mentioned previously, the processing power and flexibility of an FPGA is a good fit for SCA radio systems. However, the downconverter IP core, for example, must be software downloadable.

How the FPGA is connected to the rest of the system needs to be flexible and configurable, as in the previous downconverter example. Since the FPGA is directly in the data flow path, it can’t be a bottleneck to the high-speed of these applications.

**Non-CORBA DSP and GPP Resources**

Non-CORBA DSP and GPP resources are also considered *LoadableDevices*. GPPs can also be considered *ExecutableDevices* in some applications, but here we are addressing items such as the embedded PowerPCs in a Virtex FPGA.

In the case of processors, the executable code must be software downloadable—which should seem obvious. Any peripherals needed to support the processor must be present.

Again, in the case of Virtex FPGAs we’d expect to see the built-in processor core supported by external memory, a boot ROM, interrupt connections for some of the critical hardware, and programming paths that can be accessed so the PowerPC can be loaded—among other things.
2-Channel FPGA Transceiver for SCA

An example of a single hardware product capable of meeting the definition of the class as an SCA Device is the Pentek Model 7640 dual-channel software radio transceiver PCI board.

The Pentek Model 7640

The 7640 includes two 105 MHz 14-bit A/D converters, two 500 MHz 16-bit D/A converters, and a six million gate Virtex-II Pro FPGA that includes two PowerPC processor cores.

Other on-board resources include a four-channel digital downconverter, 512 MB of synchronous DRAM and a digital upconverter. As a PCI card, the board plugs directly into the PCI slot of a desktop PC and all communications with the board are performed over the PCI bus.

Each of the two A/D inputs can be connected directly to an RF tuner that amplifies the antenna signal and downconverts the signal frequency to IF, typically below 200 MHz.

The two D/A converter outputs can drive the IF inputs of an RF upconverter and power amplifier suitable for driving a transmit antenna. As a result, the 7640 fills the gap from the receive antenna and its receiver, to the upconverter and the transmit antenna.

Devices

Based on the device classes outlined previously, the A/D and the D/A converters of the model 7640 can be considered simple Devices, and the FPGA a LoadableDevice. Again, as mentioned previously, you might think that two PowerPC processing engines within the FPGA could be candidates for ExecutableDevices. However, it’s unlikely that all of the requirements of the operating system and CORBA structures are possible within the constraints of this specialized hardware. In fact, it’s much more reasonable that the PowerPC would be a LoadableDevice.

In this case, a C program compiled to the PowerPC could be loaded into the associated RAM and then executed upon command, analogous to the configuration code that actually loads the FPGA.
Model 7640 Applications

Some of the common applications that the Model 7640 Dual Channel Software Radio Transceiver PCI Board targets include:

- Wireless base station development platform
- Military radio transceiver development platform
- Waveform development platforms
- Communications and radar countermeasures
- FPGA IP development platform

This board is also available as a CompactPCI card or as a PMC/XMC module. These hardware form factors allow the same board to move from a development or lab environment to a field deployed platform—even in a different format.
Mission: Port a COTS Board to SCA

Imagine now that the system developer was given the task to find a COTS software defined radio transceiver and to port that board to SCA. Pentek saw this as a common development requirement and realized that we could provide developers with a tested and documented solution. The solution would effectively remove the task of porting the SCA from the developer and allow the end user to focus on the real work: developing radios and waveforms.

In the process of specifying this porting task, a flexible development platform became the ultimate goal with the following features:

- Create a JTRS radio set development platform
- Create a waveform development platform
- Include the option to make that platform deployable

As far as the platform features, the strategy was:

- Employ a transceiver to cover both ends of the radio function.
- Use a configurable platform since this is one of the requirements for SCA compliance.
- Adhere to industry standard form factors and interfaces, to allow for the move from the lab environment to a field environment.
- Utilize flexible architectures to support many different possible applications.
- Create an easily scalable system with the goal of having engineers start small but be able to expand to a large system as needed.
- Develop a low-cost system with high performance solutions.

Phased Mission Strategy

The development cycle included three major steps to complete the mission:

Phase 1

The first step was to specify and assemble a PC-based SCA software platform. This platform needed the following features:

- It had to be complete with a core framework and development tools.
- It had to include productivity tools to ease development.
- It had to include a reference hardware I/O Device.
- It had to include a reference waveform and reference applications.

Phase 2

The second step, once the software solution was found, was to create a new SCA I/O Device for the hardware,

- We decided to replace a known reference Device provided by the chosen SCA platform with a COTS board I/O.
- And, we needed to test the new Device with the reference waveforms and applications.

Phase 3

Once the new Device was in place we needed to create an example application, and part of the application was to connect the reference Device. In this case, we connected a PC sound card to the COTS board I/O.
Phase 1: SCARI++ Software Suite

After careful survey of several software vendors, we recommend the SCARI++ Software Suite from CRC (Communication Research Centre) of Canada.

SCARI++ is a full-featured C++ implementation of the SCA Core Framework. As a flexible core framework implementation, SCARI++ is completely configurable with special emphasis on ease of use.

It includes several unique features helpful for radio, introspection and application debugging. SCARI++ offers a Component Development Library (CDL) which is used to create applications and device components. The CDL actually implements the complex SCA requirements, allowing the user to concentrate on the development of applications or Devices.

For example, the CDL offers a multithread safe framework for query and configuration services, as well as for Device capacity models. These are requirements, or at least mandates, of the SCA. The CDL effectively reduces the quantity of code that developers must write and helps meet the SCA requirements.

Completing the SCARI++ Suite is the SDR Development Tool Set which is fully integrated with the CDL. It provides the developers with a comprehensive set of tools to design, run, debug, and deploy SDR systems.

◆ Component Development Library

The CDL was designed to provide an effective method of rapid application and device creation for SCA developers. It implements several common functionalities that must be part of every SCA component.

The CDL automatically handles the SCA interface through a generic implementation of the Resource interface, and it also handles the complex validations that must be performed, as required by SCA when a Resource or Device is configured or queried.

The CDL handles the mandated capacity model and other state behaviors through a generic implementation of a Device. The CDL supports the required two kinds of properties, Resource and Allocation as well as every property format: Simple, SimpleSequence, Struct and StructSequence.

◆ SDR Development Toolset

As part of the SCARI++ Software Suite, CRC developed a number of software tools to assist in the development of SDR projects. These tools are closely integrated with the SCARI++ core framework implementation to provide an even better interaction. Some of the tools included in the toolset are:

● The Component Creator Wizard is used to create all metadata files for any kind of SCA component. This GUI is form-based, simple to use and prevents the creation of invalid metadata.

[Continued on next page]
The Waveform Application Builder (WAB) is a GUI used to construct or modify waveform applications. Using the WAB, applications are created by dragging components from a toolbar into the application canvas. The WAB supports all types of connections, direct or indirect, and allows only the connections based on the component interface description. Connections are established graphically and the developer chooses the components from just a popup menu containing only compatible ports.

Components can easily be imported and organized in different toolbars. WAB can also be used to import existing application descriptors, allowing a developer to see them graphically. Finally the WAB can package an application into a single file for easy transportation and installation in many SCARI++ core frameworks.

The Radio Manager is used to control and configure an SCA radio and is the key tool for deploying the application on the hardware platform. This GUI offers a graphical block diagram of the radio nodes and applications. It displays a hierarchical view of the radio node components and each type of node component has a context sensitive menu to allow specific operations.

The Radio Manager is also used to install or uninstall applications, as well as initiate or shut down applications. Once an application has been initiated, this GUI can graphically show how each resource has been deployed, and how it is interconnected to the other resources.

Each initiated application is controlled through a generic console which displays its different configuration properties. The Radio Manager also allows custom GUIs for console control.

The details and features of the SCARI++ Suite are extensive. To fully appreciate how the SCARI++ Suite addresses SDR and SCA requirements, please visit CRC’s website:

Achieving SCA Compliance for COTS Software Defined Radio

SCA Development System

**Phase 2: A Low-cost PCI Bus Linux Solution**

The recommended development platform shown below is a complete low-cost system for developing software defined radio products.

The 7640 transceiver board that we discussed earlier, ported with all of the necessary SCA infrastructure, creates an SCA-compliant platform. Comprised of hardware and software components compatible with the SCA standard, the SCA platform is a PC-based system that runs under Linux.

The system includes an installed CRC SCARI++ core framework, Component Development Library and SDR development tools, plus the Pentek board support package for the model 7640 and all the board-specific components.

This complete platform satisfies the original stated mission by offering a low-cost solution suitable for waveform developers, application developers and system integrators.

In order to support scalability to large platforms, an additional 7640 PCI card can be installed right in the PC motherboard. Furthermore, the identical board architecture is available in a PMC form factor, or a 3U or a 6U CompactPCI form factor that can be convection-cooled, ruggedized, or conduction-cooled.

These are all extremely appropriate for large, platform-specific deployed multichannel military systems.

---

**Pentek Model SCA 2510 SDR Development System**

**CRC SCARI Development Tools**
- Core Framework
- Component Editor
- Code Generator
- Waveform Application Builder
- Node Boot Builder
- Radio Manager

**ORB TAO**
- Redhat Fedora 2 Linux
- Pentek Linux Driver for 7640 PCI Transceiver
- Pentek 7640 Board Support Package
- 1 Year Software Maintenance
SCA Application Layer

If we just overlay the COTS hardware used in this system over the SCA application layer that we discussed earlier, the components break out as follows:

The SCA Devices include:
- The A/Ds and the D/As
- The Digital Downconverter, and the Digital Upconverter
- The PC’s Audio Input and Output
Phase 3: Application Example

Having completed Phase 1 and Phase 2 of the Mission Strategy shown on page 18, we embarked on developing an example application.

As shown in the diagram below, the development example started out with a reference example provided in the SCARI++ Software Suite. This example application does echo processing on a signal received from the PC’s sound card input and then outputs the process stream to the PC sound card output.

The next step was to take the newly created Device, the Model 7640, and use it to replace the PC’s audio input with it. As part of this application, we added additional processing in the form of FM demodulation.

The next steps consisted of replacing the PC’s audio output with the 7640 transmit hardware, and then creating a full transceiver example using the 7640 from input to output.

Concluding Remarks

Pentek developed the Model 7640 transceiver as a product for people who have to develop SCA-compliant SDR. With it, you can create a custom development system in-house by purchasing the 7640 and support software from Pentek and the SCARI++ Software Suite from CRC. A Pentium 4 PC running Linux, from the vendor of your choice, completes your development system.

One topic that we have focused on throughout this handbook is the role of the FPGA in the overall hardware. Your SCA development platform can be expanded with any of those GateFlow products to exploit the performance and flexibility of the FPGAs. It’s important to understand that a key component to fully integrate the FPGAs into the SCA is still missing as of this writing.

As a bit of background, in August of 2004 the SCA introduced a specialized hardware supplement that defined the connectivity of DSPs and FPGAs. The supplement has not achieved widespread adoption and some further work seems to be indicated to better define the interfaces between SCA and the specialized hardware. When the specialized hardware is fully accepted, it will allow one to fully utilize the flexibility of the FPGAs.
Useful Links

The JTRS Website includes definitions and goals of the JTRS program, and it has downloadable specs. You can also get the SCA spec from there.

You can go to the Pentek Website for information on the Model 7640, complementary products, and support software.

For information on the CRC SCARI++ tools and core framework, go to the CRC Website.

For FPGA technology examples visit the Xilinx Website and you can also visit the Pentek Website. We have a number of white papers available on FPGAs and integrating the FPGA technology into SDR systems.

### JTRS Program
Click here: http://enterprise.spawar.navy.mil/body.cfm?type=c&category=27&subcat=60

### SCA Specification Release 2.2.2
Click here: http://jtrs.spawar.navy.mil/sca/home.asp

### SCA Development Tools

### FPGA Technology
Click here for Xilinx: http://www.xilinx.com

Click here for Pentek: http://www.pentek.com/go/gateflowhb

### Pentek Board Support and Models 7140, 7240, 7340, 7640
Click here for ReadyFlow: http://www.pentek.com/go/readyflowhb

Click here for 7140: http://www.pentek.com/go/7140hb

Click here for 7240: http://www.pentek.com/go/7240hb

Click here for 7340: http://www.pentek.com/go/7340hb

Click here for 7640: http://www.pentek.com/go/7640hb