PERFORMANCE OPTIMIZATION OF REMOTE SCA LAUNCHER

David Murotake, (SCA Technica, Inc., Nashua NH, USA; dmurotak@scatechnica.com)
Jeff Jussel (Celoxica, Inc., Campbell CA, USA; jeff.jussel@celoxica.com)

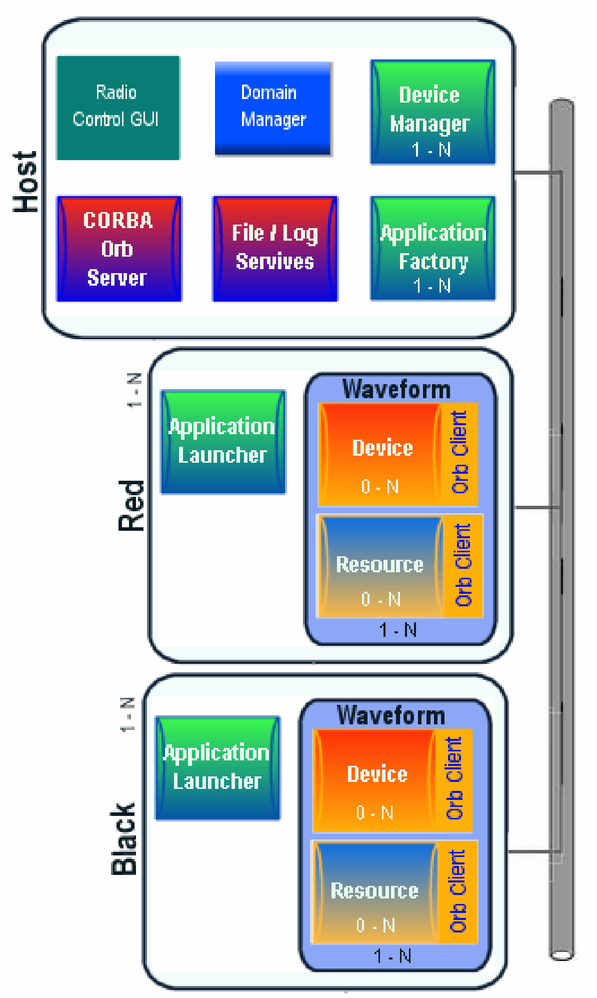
ABSTRACT

The United States Department of Defense has specified a CORBA based Software Communications Architecture (SCA) as part of a procurement specification for the family of Joint Tactical Radio System (JTRS) Cluster radios. The JTRS Cluster radios are a family of software defined radios (SDR) requiring a high-assurance, secure architecture employing NSA Type 1 security and other stringent security architecture requirements [1].

In this paper, we discuss plug-and-play versions of SCA launchers which are combined with a self-booting system usable in scalable, plug-and-play, embedded radio systems, such as SCA modem cards used with PDA and laptop computers (Figure 1). We further examine high assurance implementations which include components embodied in hardware for security and performance acceleration purposes, and software design tools for SDR which may be usable to achieve these implementations.

1. INTRODUCTION

In SDR03, Murotake et al presented “A lightweight software communications architecture (SCA) launcher implementation for embedded radios” [2] (Table 1) in which we described the use of a remote waveform launcher to dramatically reduce SCA run-time memory footprints and processing required in the embedded radio component of networked SCA systems. The launcher can be employed when distributed radio components, such as a plug-and-play modem card and transceiver, are networked with a remote host computer. The SCA components are partitioned across the networked system, and the majority of the SCA resides on a remote host, such as a laptop computer or personal digital assistant (PDA).



2.2 Launcher and a simple demonstration waveform [2].

Figure 1. Application Launcher for plug-and-play SCA red/black embedded radio system with host [2].

The remote host is less processor- and memory-constrained than the embedded radio components, and thus more capable of hosting the SCA core framework

Table 1. Run-time memory requirements (MB) using SCA

	ORB	SCA	Launcher	Resources	Total
RED1, Linux 128M Ram	0	0	6.97	8.83	**15.80**
BLACK1, XP-pro 512M Ram	0	0	9.05	12.77	**21.82**
HOST, Linux 1024M Ram	7.05	48.22	0	10.84	**66.10**
System total					*103.73*

(CF), operating environment (OE) and waveform resources. When implementing the launcher, the following security architecture factors should be considered:

- Alves-Foss et al. (2004) showed that light weight implementations (those with reduced memory and processor footprints) for high assurance systems, which must resist multi-disciplinary threats to platform integrity, optimally employ a multi-layered, defensive architecture spanning hardware and software components [3].
- Murotake (2004) further demonstrated that threats to SDR platform integrity included five major forms attack forms across both the radio and computer interfaces of the SDR.
- The SDR Forum report DL-SIN on security considerations [4] further recommends a security reference architecture including employment of a secure hardware kernel for increased resistance against attack.

A functional prototype of the high assurance SCA launcher component is under development at this time. The ongoing trade studies include finalization of launcher architecture and methods (including tools) of deploying hardware enabled components.

2. LAUNCHER SYSTEM ARCHITECTURE

A Remote Launcher is a device/server that resides on a remote client and is separate from the SCA Core Framework. It provides mechanisms to allow an application to be started on a processor or system that is remote from the heavy core framework. This requires that all needed files for the application and all libraries must either exist on a Lightweight client or the Launcher must provide a means of transferring and saving the files. The remote launcher represents the only overhead to the lightweight client that is beyond the need to run the required applications. All SCA functionality resides at the host and does not need to be replicated for each Core / Red / Black processor. The only running processes on the embedded radio cores will be the required waveforms and a very compact Launcher. The ORB and all other SCA components will operate on the host processor and memory. Unless the JTRS JPO wishes to standardize "plug and play/hot swap" operation for SCA moduem and radio muddles, no changes to the SCA 3.0 are required to incorporate, or operate, the SCA Launcher [2]. The prototype module incorporates industry standard PnP and hot swap specifications. Interface standards used are the Personal Computer Memory Card Industry Association (PCMCIA) CardBus 32 interconnect to the host laptop computer, and either Universal Serial Bus version 2.0 (USB 2.0) or 1X Express interfaces to the RF/IF module using a new industry standard dual-wireless interconnect standard proposed by the PCI SIG.

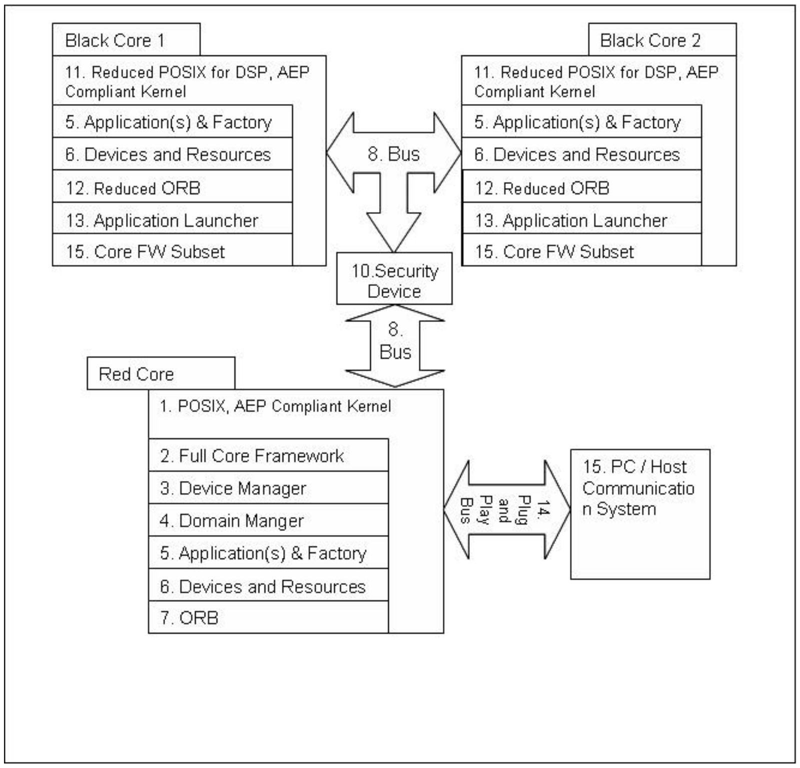


Figure 2. SCA Launcher system architecture within-line security device (patent pending).

The SCA launcher components, and allocations to the embedded Red and Black subsystems, are shown in Figure 2 for a plug-and-play SCA modem module. The Red Core, Security Device, and Black Cores comprise the PnP SCA module. The host device used in the prototype is a Trusted Computing Platform Association (TCPA) compliant, Common Criteria (CC) Evaluated Assurance Level 4 (EAL4) laptop computer by IBM running Microsoft Windows XP Pro, Service Pack 2. An embedded operating microkernel, complying with the SCA applications environment profile (AEP) reduced POSIX profile for DSP is employed in the embedded black and red subsystems. Production units are expected to use EAL 5+ components, with a goal of EAL 7.

3. ASSURANCE ARCHITECTURE

The optimal approach to defeating a blended attack using a lightweight system is a "multi-layered" defense, or defense in depth [3]. That is, a combination of methods, instantiated in both hardware and software, is implemented in both the terminal and access point, in both design and verification of high assurance systems. In the most secure high assurance systems, a hierarchical architecture is employed, where multiple layers provide specific, well-defined security mechanisms that can be used by higher levels.

A high assurance security mechanism must be: (i) always invoked, (ii) non-bypassable, (iii) tamperproof, and (iv) verifiable. HAWCS incorporates the security features recommended by the SDR Forum [5]:

1. Security Policy Enforcement and Management
2. Information Integrity
3. Authentication and Non-repudiation
4. Access Control
5. Encryption and Decryption Services
6. Key and Certificate Management
7. Standardized Installation Mechanisms
8. Auditing and Alarms
9. Configuration Management
10. Memory Management
11. Emissions Management
12. Computer Security (virus scanning applications and firewalls)

The SDR Forum's Security Reference Architecture suggests use of a hardware security module and secure download, storage, installation and instantiation (DSII) of waveforms and other applications.

4. FPGA IMPLEMENTATION TOOLS

In addition to high-speed modem resources, HAWCS deploys selected SCA launcher kernel functions to a FPGA for purposes of enhanced security and performance acceleration. These functions include:

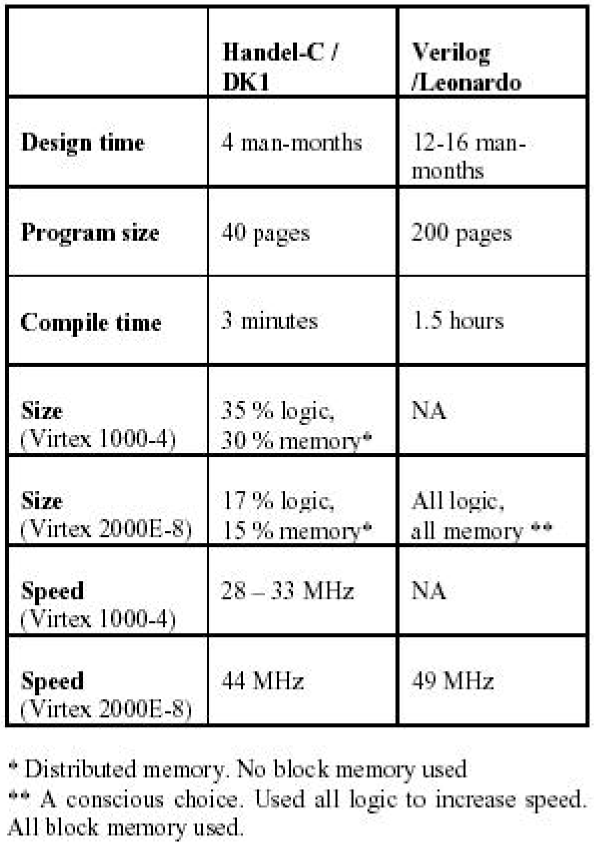
- Secure system executive micro-kernel.
- Secured programming gateway for DSII.
- In-line firewall implementing NAT and SPI.
- ORB components requiring acceleration.
- IP stack components requiring acceleration.
- I/O transport functions requiring acceleration.

Since an NSA Type 1 compatible encryption engine (the Advanced Infosec Machine, or AIM) is being employed in the project, an encryption engine is not incorporated.

Several modern FPGA co-design tools are available to the project to accelerate development of FPGA deployed functions. Examples of these software tools include:

- Mathworks MATLAB™ and SIMULINK™ and targeted co-design tools for Xilinx and Altera FPGA's.
- Accelchip™ co-design tools (MATLAB™ to VHDL). This tool will be employed to bring MATLAB waveform code to prototype implementation on the modem FPGA.
- Celoxica™ co-design tools (C to VHDL). This tool will be employed to implement the security kernel, accelerated ORB components, accelerated IP stack components, and accelerated I/O components in FPGA.

Since our waveforms are prototyped in MATLAB™, and our core framework (CF) and ORB components are written in C and C++, our project team will employ and evaluate the Accelchip™ and Celoxica™ tools during this project. However, preliminary study reveals the use of these tools may be extremely productive during our prototype phase. An example Celoxica tool use on an IPv6 acceleration project, comparable to one of our tasks, is shown in Table 2 [6].



	Handel-C / DK1	**Verilog /Leonardo**
Design time	4 man-months	12-16 man-months
Program size	40 pages	200 pages
Compile time	3 minutes	1.5 hours
Size (Virtex 1000-4)	35 % logic, 30 % memory*	NA
Size (Virtex 2000E-8)	17 % logic, 15 % memory*	All logic, all memory **
Speed (Virtex 1000-4)	28 – 33 MHz	NA
Speed (Virtex 2000E-8)	44 MHz	49 MHz

* Distributed memory. No block memory used
** A conscious choice. Used all logic to increase speed. All block memory used.

Table 2. IPv6 Protocol Stack Accelerator Results. [6]

In this project, two parallel groups of Ericsson designers used the same specification to implement the same functionality, one using traditional hardware design methods and tools, while the second group employed the Celoxica™ co-design tools which employs an ANSI C-like language called Handel C. The co-design application programming interfaces (API) for the Celoxica Data Streaming Manager (DSM) are shown in Figure 3. The design effort implemented a hard-real-time table-lookup for Internet Protocol packet forwarding in an IPv6 router. The implementation also included design of a memory interface and a simple S-Bus interface for interfacing the module with a host processor. An implementation of Internet Protocol Header Compression for a hardware-based Internet Router was also done.

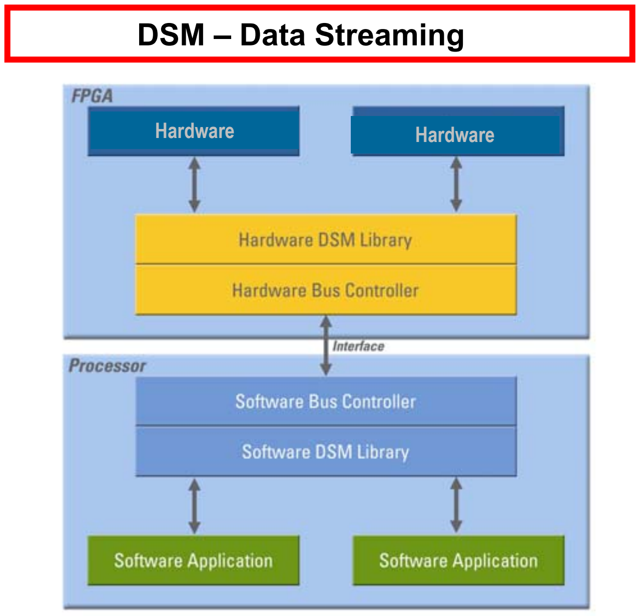
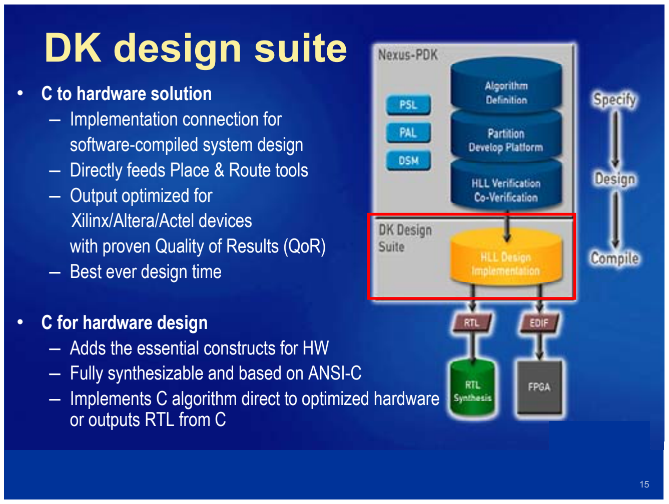


Figure 3. Celoxica™ co-design API's (C to VHDL)



DK design suite

- C to hardware solution
  - Implementation connection for software-compiled system design
  - Directly feeds Place & Route tools
  - Output optimized for Xilinx/Altera/Actel devices with proven Quality of Results (QoR)
  - Best ever design time
- C for hardware design
  - Adds the essential constructs for HW
  - Fully synthesizable and based on ANSI-C
  - Implements C algorithm direct to optimized hardware or outputs RTL from C

Figure 4. Celoxica™ design suite for design, solutions.

In another design example, a DSP acceleration project targeting a 2 million gate, 96 MHz Altera FPGA was projected to require 25 man months with engineering costs of $1.6 million. By contrast, a parallel effort using these design tools was completed in only 4 man months at a total project cost of $200,000.

The design suite has components to support both hardware design and hardware solutions, including targets for Xilinx, Altera and Texas Instruments target platforms (Figure 4). Using the tools, designs can be completed 3-6 times faster than traditional methods, with similar results in terms of speed and area.

The more rapid implementations can be attributed to:

- Support for sequential logic. Sequential models can be parallelized more easily than sequencing a parallel model.
- Compact representation in the C language.
- Software-like design methodology, with debuggers, with fast turnaround in design environment

This acceleration of project schedules and more efficient use of gates within the FPGA for comparable performance acceleration, are exceptionally promising results.

5. CONCLUSIONS AND FUTURE RESEARCH

SDR security is a system level problem. Lightweight SCA compliant systems must not only protect classified data (COMSEC) or protect signals using frequency hopping or spread spectrum methods (TRANSEC). SDR must also protect the integrity of the programmable radio platform against wireless and Internet hackers using blended attacks. To defend against the blended attack requires a multi-layered defense-in-depth which protects both the mobile terminal (radio set) and network servers.

Previously, FPGA's were primarily considered a means of accelerating time-critical modem digital signal processing functions, such as the bit-serial processing needed in spread spectrum receivers and transmitters. However, it is now apparent that other functions of a high assurance, SCA compliant modem should be committed to hardware. These include key security functions which require the additional security of hardware implementation, and certain time-critical SCA functions such as I/O and middleware functions which require acceleration to provide adequate performance.

Co-design tools based on translation of C, MATLAB™ and SIMULINK™ software to VHDL offer great promise to rapid prototyping and system development of SDR components, such as the self-booting, PnP SCA launcher being developed by our project team. As we employ these tools in developing the hardware accelerated, self-booting SCA launcher over the next year, we plan to publish our results in a future paper to the SDR Forum.

6. REFERENCES

[1] Joint Tactical Radio System (JTRS) Joint Program Office, "Software Communications Architecture Specification", JTRS-5000, SCA V3.0, August 2004

[2] Murotake et al, "A lightweight software communications architecture (SCA) launcher implementation for embedded radios", SDR Forum Technical Conference, Phoenix AZ, November 2003

[3] Alves-Foss, Taylor, and Oman, "A multi-layered approach to security in high assurance systems", Proceedings of 37th Hawaii International Conference on System Sciences – 2004".

[4] Murotake, "System Threat Analysis Case Study for Software Based Communications", OMG Software Based Communications Workshop, September 2004

[5] "Security considerations for operational software for software defined radio devices in a commercial wireless domain", SDR Forum DL-SIN, Document SDRF-02-W-0005-V030, October 2004.

[6] Torkelsson and Ditmar , "Header Compression in Handel-C - an Internet Application and a New Design Language", Ericsson Radio Systems AB, Sweden, August 2001.

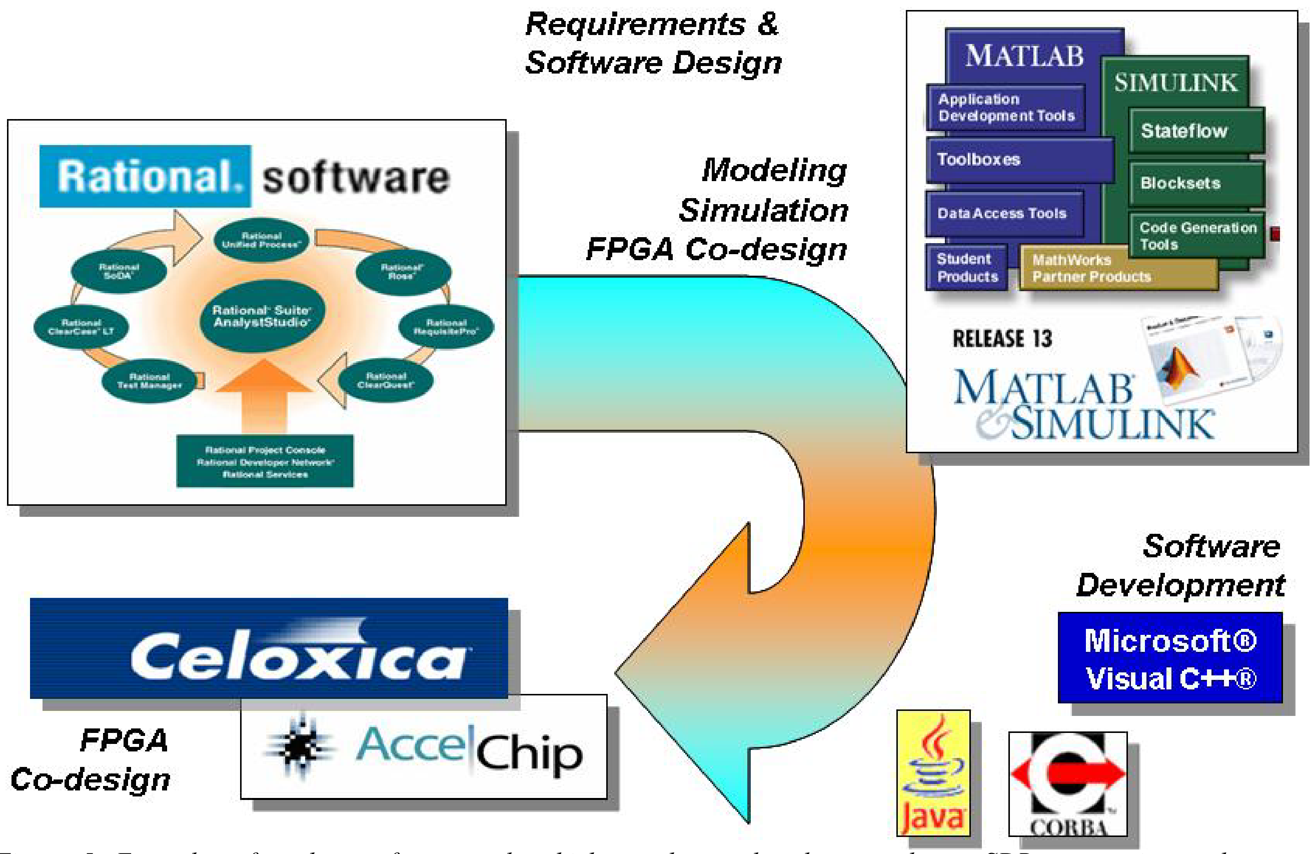


Figure 5. Examples of modern software tools which can be employed to accelerate SDR prototyping and system development.