DEVELOPING AN SCA GSM WAVEFORM TARGETED ON A DSP/FPGA ARCHITECTURE

Louis Bélanger (Lyrtech inc., Quebec, Quebec, Canada louis.belanger@lyrtech.com)

ABSTRACT

High-level “frameworks” such as the SCA (Software Communication Architecture) add virtualization layers on heterogeneous DSP-FPGA systems with the promise of code portability., However, there is an impact on performance and complexity. This paper takes a look at these aspects by using a design example consisting in the design of a GSM waveform compliant with the SCA framework. More specifically, this paper takes a look at the lower level detail aspects of developing the IF and baseband processing components of the SCA waveform on a FPGA and a DSP, respectively Moreover, the project is executed using the model-based/system-level development tool MATLAB®/Simulink® to improve testing and productivity, and bridge modeling and implementation phases

First, a review of the basic technical aspects of GSM is given. Emphasis is placed on IF and baseband aspects. Then, elements of SCA which are in the scope of this implementation are summarized. A brief presentation of development tools and design flow then follows. Lastly, results of GSM waveform implementation are presented, followed by a conclusion and an indication of future works.

1. PROJECT CONTEXT

The implementation was done in the general context of the development of an SCA board support package for Lyrtech’s DSP/FPGA development platforms as well as the development of example applications and reference designs running on these platforms. In this context, these platforms provide a complete SCA waveform development environment at the DSP/FPGA level and help JTR radio developers address the development challenges of the promising but rather complex SCA environment.

As far as GSM is concerned, the scope is to have a basic capability system that operates as a GSM system while demonstrating a typical SCA implementation at the DSP/FPGA level, in line with the proposed SCA extensions related to DSP specialized hardware (due for inclusion in an upcoming revision of the actual version 2.2 [1] of SCA). The target platform is the family of SignalMaster™ DSP/FPGA platforms. IF processing is performed in the FPGA and its design is developed in System Generator™ in a system-level environment. In a complementary manner, baseband processing implementation is performed in the DSP, using a similar Simulink® Design Flow with the Real-Time Workshop® C-code Generator, the Embedded Target for TI DSP Toolbox and Lyrtech’s GSM DSP libraries. SCA-related elements are added to wrap the executable code and provide SCA-defined portability and scalability capabilities, which is also in line with the recently proposed IIM (implementation independent model) & ISM (implementation specific model) SCA waveform model-based design flow,.

2. GSM PROCESSING

Figure 1 presents a GSM physical channel. There are 124 × 200 kHz channels that are frequency-multiplexed in a 25 MHz-wide RF spectrum, one for each downlink and uplink path. Figure 1 also shows in more detail how the “bursts” of each GSM channel are constructed. Basically, each burst is part of an 8 slot TDM frame, forming a 200 kHz wide spectrum. Each burst has tail bits and an extended guard interval to avoid interference as long as the MS (mobile station) is within 35 km of the BS (base station). Some fixed training-bit sequences allow synchronization between the MS and the BS.

Use of the GSM application as a design example corresponds very well to our platform’s segmented architecture. In Figure 2, the main uplink physical layer elements of a GSM speech and data transmission chain are presented. Figure 2 also illustrates how the processing can be partitioned. IF processing is performed on the FPGA with functions such as polyphase DDC (digital down converter), DDS (direct digital synthesis), and GMSK (Gaussian minimum shift keying) modulation. DSP-based baseband processing can tackle the tasks of encoding, encrypting, and interleaving, as well as burst building functions. Finally, communication protocol handling is performed at the RISC processor level (or in the DSP for simplified protocols, such as in our case).

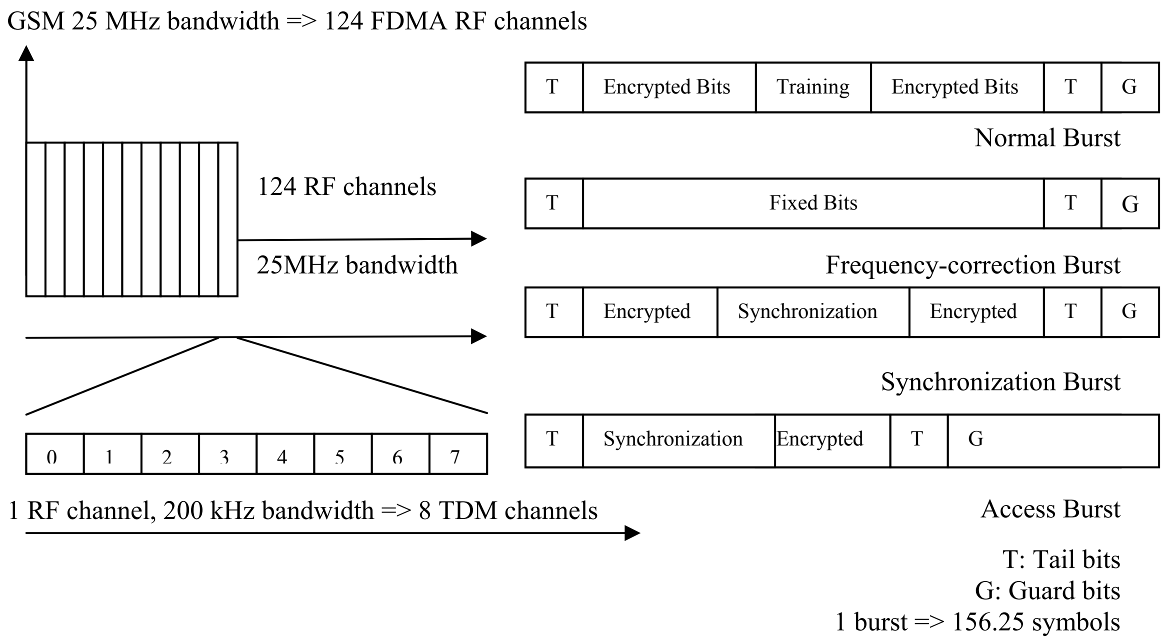

Figure 1: GSM FDM, TDM and burst structure

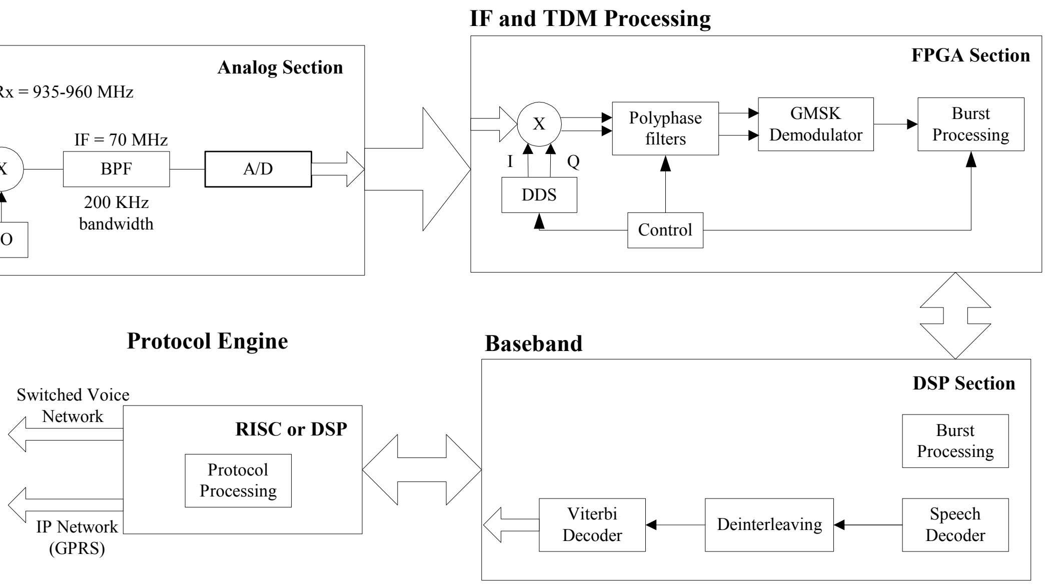

Figure 2: Partitioning of the GSM processing between the FPGA, the DSP, and a RISC processor (note: an analog RF front-end is also shown for the sake of completeness)

3. GENERAL SCA ARCHITECTURE CONCEPTS

The JTRS SCA architecture, also promoted by the SDR Forum, is a modularized architecture for wireless systems based on POSIX and OO (object-oriented) software technologies that can run and map on processing hardware ranging from a homogeneous GPP (General Purpose Processor) environment to a heterogeneous and distributed environment, which is more or less the norm with current radio or base station implementation. The potential benefits of such an approach are numerous:

- Segmentation of the "problems";
- Scalability
- Multi-vendor offering;
- Decoupling of technology advances on specific functions (RF, IF, baseband)
- Enhanced software portability

The generic SDR architecture, shown in Figure 3, consists of functions connected through open interfaces, and procedures for adding software-specific tasks to each of the functional areas. The software necessary to operate is referred to as an application waveform. Figure 3 shows an open architecture of seven independent subsystems interconnected by open interfaces. Interfaces exist for linking software application specific modules into each subsystem. Each subsystem may contain processing hardware, firmware, an operating system, and software modules that may be common to more than one application. The application layer is modular, flexible, and can be mapped to different hardware. The common software API layer is standardized with common functions having open and published interfaces, based on OMG (Object Management Group) and IDL (Interface Definition Language) standards.

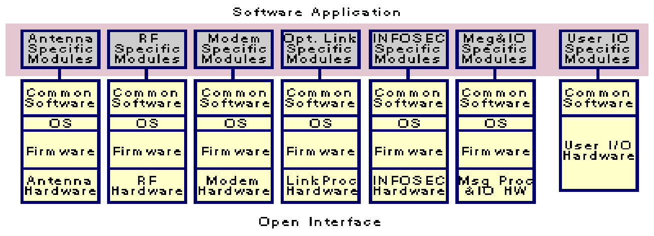

Figure 3: Typical implementation of SDRF software and hardware open architecture

4. SCA DSP-ORIENTED SPECIALIZED HARDWARE SUPPLEMENT (SHS)

The more recently-published Specialized Hardware Supplement (SHS) [2] provides more precise guidance to address portability of software for processing elements other than general purpose processors.

The SHS supplements the SCA specification by specifically addressing the software for specialized hardware: field programmable gate arrays (FPGA), digital signal processors (DSP), and ASICs. In general, a software application of this type runs on a collection of specialized hardware components that are connected via special purpose data buses. The SHS supplement specifies:

- A hardware abstraction layer connectivity standard (HAL-C),
- A reduced POSIX AEP for DSP environments,
- Standard waveform functional blocks to be provided as part of each platform

The requirements of this supplement are in fact intended to mitigate a set of problems that reduce the cost of portability for this type of software. These problems include the lack of standard operating systems in DSPs and the differing computational paradigms of DSPs, FPGAs, ASICs, network processors and other special function devices.

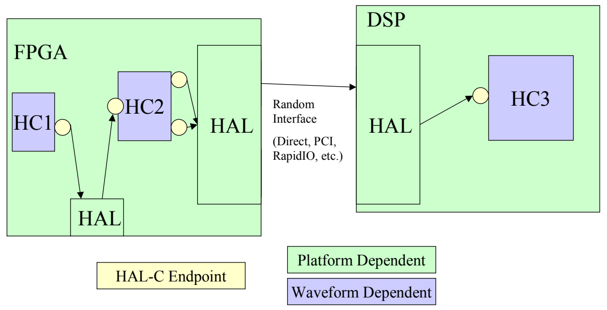

Figure 4: HAL-C DSP/FPGA model

Figure 4 shows the implementation model of HAL-C. A HAL-C realization consists of both waveform- and platform-specific components. A HAL-C Component (HC) implements functionality required by a waveform. These components are written by the waveform developer and designed to the HAL-C API so that their portability potential is maximized. To this extent, components are interconnected using the concepts of source and sink ports, adapted to DSP and FPGA environments from the more generic interfacing guidelines of IDL.

Figure 5 displays the port interface definitions for FPGA. Again, industry standards have been used to inspire this definition, which is based on OCP (open core protocol). A software-based equivalent of these ports exists for the DSP side.

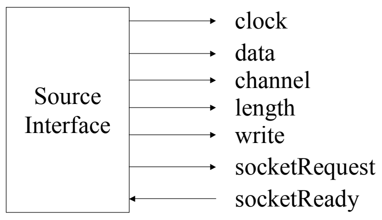

Figure 5: OCP-inspired FPGA-located source interface port definition

5. OS SERVICE APIS AND OO FOR DSP ENVIRONMENTS

The SHS defines common DSP-oriented operating system (OS) services, based upon a POSIX AEP (Application Environment Profile) subset extracted from the SCA POSIX AEP. These common services can be implemented across all SCA Digital Signal Processing (DSP) implementations. Also, the features of HAL-C were considered in the selection of functions that support interprocess communication.

In addition to OS services, object-oriented approaches might also have to be used in the DSP environment (but not in the FPGA), to conform to the SCA paradigms (an alternative and more classical approach is to run an SCA DSP device driver on a GPP in a proxy manner). While one can argue about whether or not to introduce object-oriented programming concepts in a DSP context, where speed and code size are critical, modern DSP development systems are becoming more and more OO-aware and ready. As an example, the XDAIS TMS320 DSP Algorithm Standard™, proposed by Texas Instruments, closely follows the object-oriented paradigms.

In fact, DSP algorithms are quite suitable candidates for object oriented programming as they exhibit a common behavior in that they all act as data transducers that convert the input data stream into an output data stream by applying a suitable transform on them.
The XDAIS component model allows algorithms to be created in a structured manner and uses the following object oriented characteristics:

- Abstraction
- Encapsulation
- Polymorphism
- Inheritance

In this type of OO-ready DSP environment, the implementation of HAL-C concepts such as sink and source ports becomes easier, as well as integrating XDAIS-compliant DSP library components.

6. SCA WAVEFORM MODEL-BASED DESIGN FLOW

The SCA community is encouraged to adopt advanced design flow techniques [3], which emphasize model-based design. Essentially, the model-based approach, also referred to as system-level, resides in leveraging as much as possible the same environment used in the early simulation and design phase throughout the implementation phases, such as floating-point to fixed-point conversion, in-circuit testing (HIL) and verification of real-time hardware, thus improving the overall quality of the design process. A key element of this flow resides (optionally) in automatically generating execution code from a simulation model.

The SCA uses the terminology of implementation independent models (IIM) and implementation specific models (ISM). IIM models will run on host-based GPP computers, while ISM models will represent various versions of this model implemented on heterogeneous hardware including the As-built waveform.

Non-field system related components such as the channel model, for example, can also be segmented, hardware targeted and run as a real-time test bench to test the final target hardware. The overall idea here is to provide some kind of "expertise continuum" between the simulation phase of a project and its field-testing phase. In this way, errors noticed in the field can be identified and corrected prior to real field testing.

To summarize, the use of a model-based simulation and implementation approach allows users to:

- Co-simulate model components in the host computer (IIM) and target hardware (ISM)
- Perform HIL (hardware-in-the-loop) verification of specific components of the model,
- Execute in real-time specific model components, with system-level monitoring, testing and verification.

7. HF SSB RADIO EXAMPLE

The following example of a typical military radio, an HF SSB AM radio, showcases the different steps involved in a model-based design process. The first step is to simulate, at the system level, the function of the chosen radio and to select a processing architecture capable of executing in real-time the code generated.

The first step in the process is to create an implementation-independent model. This can be accomplished using the Simulink® DSP blockset. Figure 6 shows the general signal flow in the radio. The local oscillators are shared by both receiver and transmitter processing chains. The receiver consists of a digital down converter followed by a final filter-demodulator stage. The transmitter is simply the receiver blocks turned around with interpolating rather than decimating filters. Once satisfactory simulation results have been obtained, indicating that the general signal flow and filtering are correct, it is time to split the model between the FPGA and the DSP. For this design, partitioning the functions is straightforward.

A tendency in "system-level development", in order to compensate for the inefficiency of "generic" automatic code generation with respect to manually coded applications (either at the C or assembly levels), is to combine the use of

C-code based simulation with optimized libraries or cores substituted when building the target application. Such an approach is best exemplified in the Xilinx® System Generator™ extension to Simulink. In this case, "target specific" structural VHDL is generated, as a compromise to "pure" system level code.

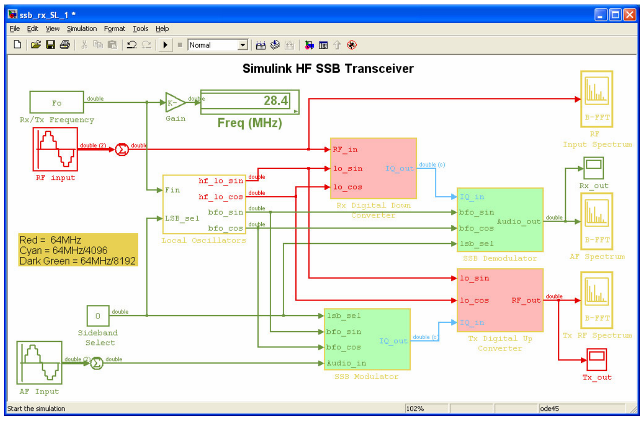

Figure 6: This top-level model captures the five main subsystems needed to create the SSB radio. Each block contains several levels of hierarchy, getting down to fixed-point implementation details where required. The light red blocks handle the 64 Msps data rates and will therefore be implemented in the FPGA. The green blocks handle data at both 15.625 and 7.8125 Ksps, and will be implemented in the DSP chip.

8. SIGNALMASTER PLATFORM FAMILY

Lyrtech Signal Processing offers a complete range of Wireless-capable DSP/FPGA platforms, ranging from the single-channel (65 MHz) SignalWAVe, to the dual-channel (2 x 105 MHz) capable SignalMaster, the quad-channel (4 × 105 MHz) Quad 6713-6416/VII platform and the 8-16-32 channel (105 MHz) FPGA-based VHS series. Entry-level platforms such as the SignalWAVe, which includes a GPP Pentium-class processor for protocol processing and Ethernet-based communications, can be used as a heterogeneous GPP/DSP/FPGA SCA embedded platform, while multi-channel systems can be used for advanced Smart antenna/MIMO multi-channel system development. The platform comes with numerous example applications, including the HF/VHF SSB AM radio described above. More complex example systems such as GSM and MIMO applications are also available. Lyrtech is also in the process of developing SCA board level support for these products, accompanied by SCA Waveform demo applications. These applications will operate across different platforms, demonstrating SCA cross platform portability and scalability, as well the SCA model-based IIM and ISM design flow.

9. GSM SCA IMPLEMENTATION

The SCA GSM implementation was derived from a non-SCA model-based GSM design. Figure 7 shows the GSM baseband model as targeted at the DSP, which can be described as an ISM model in SCA terminology, since it incorporates an implementation-specific DSP/FPGA segmentation of processing load. Note that in this model, upper layer protocol software runs on the DSP as well. The reader can find more information about this model in [5].

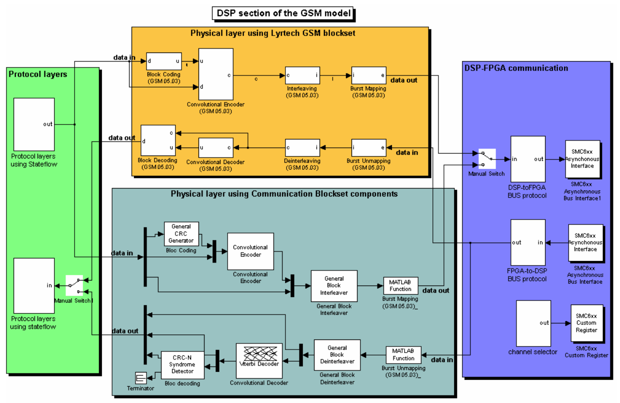
DSP section of the GSM model
Physical layer using Lyrtech GSM blockset
Protocol layers
Protocol layers using Stateflow
Block Coding (GSM 05.03)
Convolutional Encoder (GSM 05.03)
Interleaving (GSM 05.03)
Burst Mapping (GSM 05.03)
data in
data out
Block Decoding (GSM 05.03)
Convolutional Decoder (GSM 05.03)
Deinterleaving (GSM 05.03)
Burst Unmapping (GSM 05.03)
Physical layer using Communication Blockset components
General CRC Generator
Bloc Coding
Convolutional Encoder
General Block Interleaver
MATLAB Function
Burst Mapping (GSM 05.03)
CRC-N Syndrome Detector
Bloc decoding
Viterbi Decoder
Convolutional Decoder
General Block Deinterleaver
Burst Unmapping (GSM 05.03)
Terminator
Manual Switch
DSP-FPGA communication
DSP-to-FPGA BUS protocol
SMC6xx Asynchronous Bus Interface
FPGA-to-DSP BUS protocol
channel selector
SMC6xx Custom Register

Figure 7: DSP Model of baseband GSM processing with comparative host/target blocksets

Building on these models, we introduced SCA sink and source ports. In the case of the FPGA model, these ports consist of special purpose VHDL code wrapped in a System Generator block. This allows GSM modem subfunctions to become encapsulated with the VHDL SCA sources and sinks and to become more portable across FPGA devices of the same class.

Figure 7 shows this model and the associated IF modulator (Figure 8) component, which applies a frequency shift on the signal supplied by the port 'Sink data' (the shifting frequency is determined by the input port 'Sink chan_sel'). One can see how the IF modulator has been encapsulated with sink and source ports to become an SCA-compliant FPGA-based component. The overhead created by the sink and the source ports is rather minimal, moreover when the components are co-located in a single FPGA "container" (using terminology defined in the SCA).

With these sink and source wrappers, the SCA Application Factory, responsible for mapping the JTR radio resources to the waveform application processing requirements, will then be capable of assembling FPGA components of waveforms in different radio-specific FPGA implementations.

The DSP side will also consist of sink and source ports implemented, however, in software. Again, the overhead is rather minimal, since these ports look like standard DSP-coprocessor drivers, especially if underlying DMA mechanisms are used.

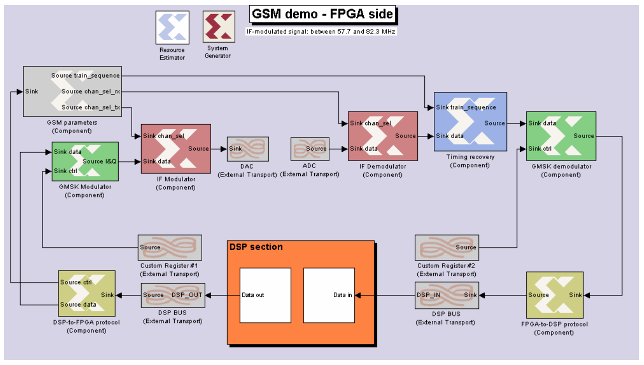

Figure 7: FPGA model with SCA-defined elements

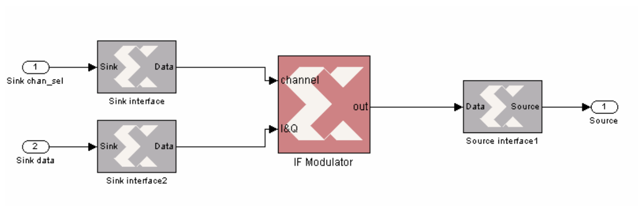

Figure 8: IF modulator function with SCA sink and source wrappers

10. CONCLUSION

This paper presented how a GSM waveform can be implemented for an SCA environment on a DSP/FPGA architecture. First, a short review of GSM concepts was presented. Then, the generic SCA architecture was briefly described, followed by a more in-depth description of recently introduced SCA concepts, such as a precise DSP/FPGA level of representation and subsets for DSP operating systems, as well as IIM and ISM design flows. A typical Simulink model-based design flow was presented, and then shown applied to a GSM DSP/FPGA model incorporating SCA-related concepts. The overhead created by these ports was mentioned as being modest. All these recent advances in SCA technology may mean that SCA is getting more and more ready for prime-time use in all military and commercial wireless systems and devices, thus realizing the original SDR vision of truly reprogrammable radios and portability across all radio platforms.

REFERENCES

[1] Joint Tactical Radio System (JTRS) Joint Program Office (JPO), *Software Communication Architecture Specification, JTRS-5000SCA* V2.2.1, April 30, 2004

[2] Joint Tactical Radio System (JTRS) Joint Program Office (JPO), *Specialized Hardware Supplement to the JTRS Software Communication Architecture (SCA) Specification, JTRS-5000 SP*, V3.0, 09 July 2004

[3] Joint Tactical Radio System (JTRS) Joint Program Office (JPO), *Acquisition Guidance to the JTRS Software Communication Architecture (SCA) Specification, JTRS-5000 SP*, V3.0, 30 June 2004

[4] Joint Tactical Radio System (JTRS) Joint Program Office (JPO), *Proposed Studies to the JTRS Software Communication Architecture (SCA) Specification, JTRS-5000 SP*, V3.0, 30 June 2004

[5] Louis Belanger, Lyrtech inc., *Development of a GSM Modem on a DSP/FPGA Architecture Using Simulink and System Generator*, *Xcell Journal*, Issue 51, Winter 2004

ABOUT THE AUTHOR

Louis N. Bélanger is a founder of Lyrtech inc., an established electronic development service company based in Quebec City, Canada and Product Development Manager of its SignalMaster DSP/FPGA signal-processing development platforms. He graduated in 1979 from Laval University's Electrical Engineering Department, and earned his MSEE in 1985 in signal processing.

Copyright Transfer Agreement: The following Copyright Transfer Agreement must be included on the cover sheet for the paper (either email or fax)—not on the paper itself.

“The authors represent that the work is original and they are the author or authors of the work, except for material quoted and referenced as text passages. Authors acknowledge that they are willing to transfer the copyright of the abstract and the completed paper to the SDR Forum for purposes of publication in the SDR Forum Conference Proceedings, on associated CD ROMS, on SDR Forum Web pages, and compilations and derivative works related to this conference, should the paper be accepted for the conference. Authors are permitted to reproduce their work, and to reuse material in whole or in part from their work; for derivative works, however, such authors may not grant third party requests for reprints or republishing.”

Government employees whose work is not subject to copyright should so certify. For work performed under a U.S. Government contract, the U.S. Government has royalty-free permission to reproduce the author's work for official U.S. Government purposes.