SMART STRATEGIES FOR INTEGRATING FPGAs AND IP CORES IN SDR SYSTEMS

Rodger H. Hosking
(Pentek, Inc., One Park Way, Upper Saddle River,
New Jersey, 07458, USA, rodger@pentek.com)

ABSTRACT

FPGAs (field programmable gate arrays) are now enjoying recognition and adoption by a wide range of software defined radio system designers and waveform developers.
While FPGAs promise dramatic increases in the achievable performance levels, delivering on this promise requires careful analysis of hardware architectures, FPGA development tools, commercial FPGA IP (intellectual property) core offerings, skill levels of engineering personnel, and techniques for waveform portability and field reconfigurability.
This paper addresses perspectives on these critical issues derived through experience gained from actual FPGA hardware implementations, development of FPGA IP for open architecture boards, and high-performance DSP algorithms.

1. INTRODUCTION

Each customer must approach acquisition of FPGA technology with his own unique perspective. Developers must not only determine whether FPGAs are right for an application, they must then also choose the best strategy for incorporating them. Since these two decisions are tightly coupled, in some cases, even though an FPGA may be ideal for an application, there may be no viable strategy for implementing the design.
Because FPGAs impose such a significant impact on system or product architecture, it is essential to establish the design strategy early in the development cycle to ensure a successful implementation from hardware selection through the development cycle and finally into deployment.
We will start with a look at a typical software defined radio architecture and then discuss methods for custom FPGA development.

2. SOFTWARE DEFINED RADIO HARDWARE

FPGAs are now being incorporated in many open-architecture software radio board level products, often referred to as COTS (commercial off the shelf). One of most successful strategies for a software radio receiver is shown in Figure 1. In the front end module, the RF input signal is digitized by an A/D converter and then delivered to a digital down converter (DDC) ASIC (application specific integrated circuit).
The A/D output and the digital down converter output both feed the FPGA. In this way, the FPGA can process the wideband output from the A/D directly, or process the narrowband outputs from the DDC. Including the DDC in the design eliminates the considerable FPGA resources that would otherwise be consumed to implement this popular DDC function.
Processed data from the FPGA is then sent to the processor board through the system interface, often a mezzanine or daughter card interface or else a backplane bus.
The inclusion of a FIFO (first in first out memory) in the data stream helps the processor aggregate blocks of data to take advantage of efficient DMA (direct memory access) transfers.

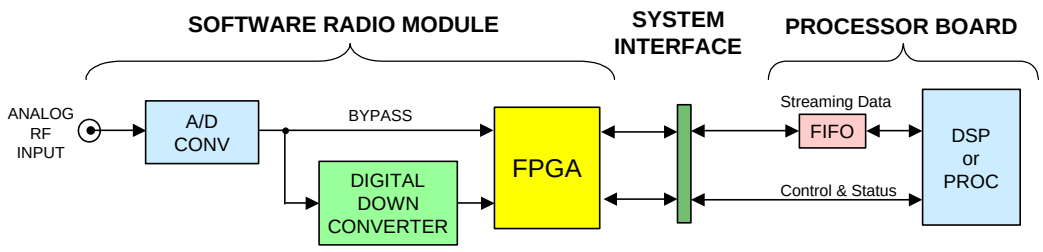

Figure 1. Typical FPGA-based Software Radio Hardware Architecture

While this architecture has proven extremely powerful, simply incorporating FPGAs in the hardware circuitry falls far short of satisfying the needs of the end user who may need to add custom features or IP cores to the FPGA. COTS board designers must carefully architect the FPGA resources so that customers can either add, delete or modify functions easily, with a minimum risk of "breaking" the basic operation of the product.

This requires significant planning during the FPGA design and development phase to carefully segment the standard factory functions so that subsequent enhancements by the customer are successful. In addition, the board vendor must create a comprehensive FPGA design and documentation package to support the customer's FPGA design environment and skill level.

3. FPGA CONFIGURATION STRATEGIES

One commercial offering to facilitate custom FPGA code development for COTS products is the Pentek GateFlow FPGA Design Kit, tailored specifically for each product and developed during the product design phase to ensure modularity of the FPGA structures.

Figure 2 shows a simplified block diagram of a generic FPGA-based software radio module showing the hardware connections to the FPGA. Each external device requires a specialized hardware interface structure (not shown) for the A/D converter, digital down converter, clock and trigger section, and the system interface.

The FPGA also includes functional blocks that implement the standard factory functions for the COTS product such as data selection and formatting, interrupt control and DDC control and status. Each of these interfaces and functional blocks is implemented as an individual VHDL configuration code module.

After all of these factory features have been implemented, a significant percentage of FPGA resources remain unused and available for customer-installed signal processing and control functions.

The Pentek GateFlow FPGA Design kit includes complete VHDL source code for all VHDL modules in the standard factory product, along with all project files, pin definition files, signal flow charts, JTAG chain definition files, and bit stream images used to create the end product.

A special VHDL module called the User Block sits directly in the data path between the Data Select and Data Formatter blocks. In the VHDL code for the standard factory FPGA configuration, the User Block implements a "straight wire" between input and output, with predefined logical pin definitions for data, clock and control signals.

This User Block was designed specifically for customers who wish to add a signal processing function to the data stream on its way through the module, such as a filter, demodulator or decoder.

In this case, the customer develops, tests and qualifies his algorithm as a stand-alone VHDL module using the FPGA design tool suite. He then drops this algorithm into the User Block, complying with the pin definitions, data flow connections and timing constraints. He then recompiles the project with all of the other unaltered VHDL modules and project files from the design kit. This approach assures a high probability of success with very little risk of disturbing critical peripheral functions.

For customers with more extensive requirements that require modification of functions outside of the User Block, full VHDL source code is provided for every other module in the FPGA. This two-tiered approach provides an easy implementation for simple applications and yet full support for major enhancements.

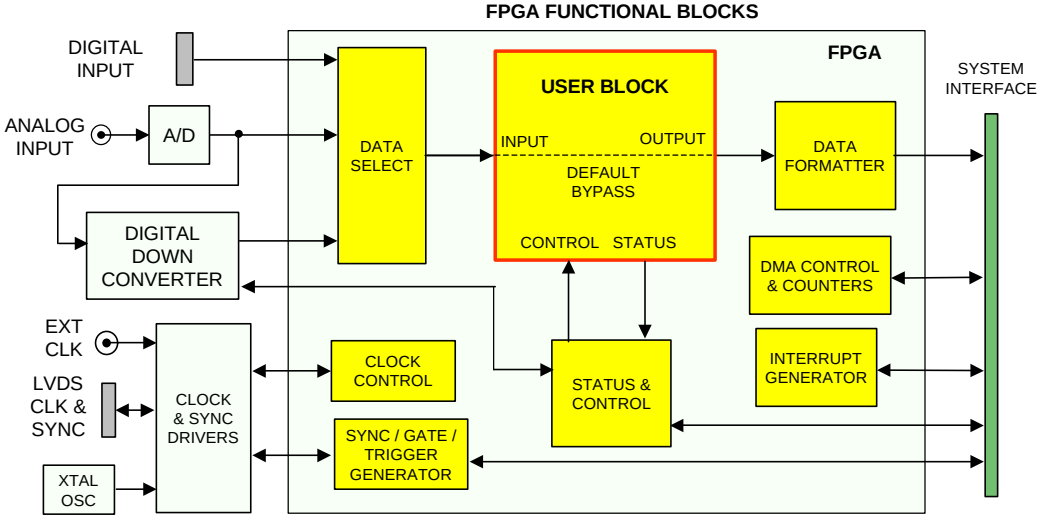

Figure 2. Typical FPGA-based Software Radio Hardware Architecture

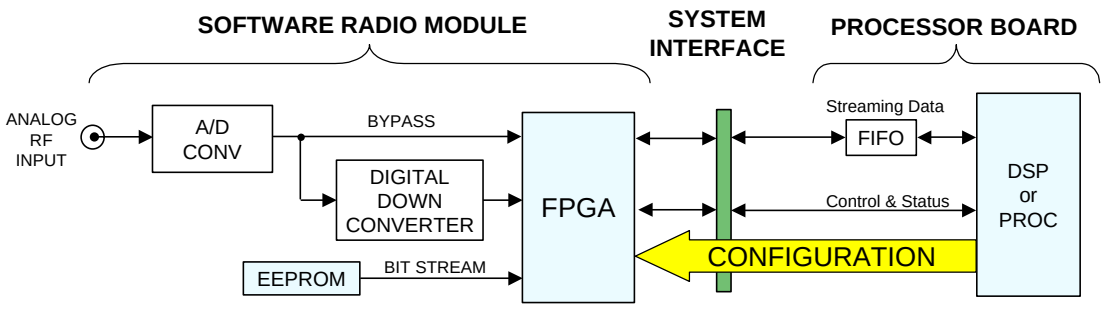

Figure 3. Configuration Code Loader Utility Simplifies FPGA Upgrades

4. FPGA CONFIGURATION CODE LOADER

For COTS FPGA-based products, the standard factory configuration code for the FPGA (often called a bit stream) is usually stored in a non-volatile serial EEPROM. When power is applied to the unit, this bit stream is copied into the volatile RAM-based cells of the FPGA to implement the desired functions.

Newly created, custom versions of the configuration code can be loaded into the EEPROM, replacing the original factory code. This usually involves disassembling the hardware and attaching a programming cable.

A more flexible alternative puts the system processor in charge of reconfiguring the FPGA. The GateFlow Design Kit includes a software module called a Loader Utility that executes on the DSP or system processor under program control. The new configuration code is encapsulated as a code module that accompanies the executable program for the system.

When invoked, the Loader Utility reads the code module and sends it across the system interface directly into the FPGA on the software radio module as shown in Figure 3. In operation, the standard factory code that had been originally loaded into the FPGA when power was applied is simply overwritten. The contents of the EEPROM remain unchanged and are available as a default configuration.

This allows easy upgrades to fielded hardware such as a JTRS radio set that needs a new waveform installed. It also allows the FPGA to be reconfigured as required by the changing operational needs of a mission.

5. INTELLECTUAL PROPERTY (IP) CORES

One of the most important resources for both hardware and software designers is the vast collection of IP core library offerings for FPGAs. Targeted for specialized functions, each represents a highly optimized structure of FPGA resources usually supplied without source code in object form. When installed in a compatible FPGA development tool environment, they appear as objects that can be dropped into the block diagram and connected into other design elements using sophisticated graphical editors. Major FPGA vendors like Xilinx and Altera offer a long list of free and licensed IP cores, but even more impressive is the emergence of a thriving new industry of third party IP core vendors. Each vendor crafts IP core offerings using the company's expertise in specific niche technologies.

Care should be taken when evaluating benchmarks for FPGA devices and the associated IP cores. Often they assume that data has already been loaded into internal block memory and that the operation is finished when the result is written back into internal memory. Getting the data into and out of these memories takes extra time and it must be taken into account.

Algorithms run faster when exception handling resources are omitted, and this can make benchmarks look deceptively fast. Designers may be able to guarantee through system architecture that certain exceptions simply cannot occur, but mysterious problems can appear in deployed systems if this aspect of the design is overlooked.

To help validate new designs, take full advantage of simulation tools. Many of the advanced simulators are now bit-true, meaning they perform the operations with exactly the same number of bits used in the FPGA hardware.

Be sure to allow enough time to thoroughly test a new FPGA design under all modes of operation to make sure it will act as reliably as the high-volume standard ASIC being replaced.

Many IP cores are parameterizable, so that designers can specify the number of bits of calculation to tradeoff space for accuracy, if necessary. Be sure that the IP cores have enough precision to meet the system requirements.

Many commercial IP cores as sold as object files without source code, so even though they may have parameterizable features, they may need to be adjusted somewhat to operate as required and to interface with other FPGA features, such as conforming to the User Block described earlier. Be prepared to modify the source code of off-the-shelf IP cores when source code is available, or else contract with the IP core vendor to make these necessary changes for you.

6. NEW FPGA DEVOLPMENT TOOLS

To facilitate custom FPGA configuration code and incorporation of IP cores, the latest class of FPGA design entry tools allows functional definition at very high levels. They include schematic capture and generator tools, HDL editors, state diagram editors, logic synthesizers, plus SPICE and IBIS modelers. Since engineers need to guarantee critical timing parameters unique to each peripheral device, specialized auxiliary tools now provide strict management of clocks, control signals, and data streams.

Recent FPGA innovations include user-configurable I/O pin drivers for multi-gigabit serial interfaces to support the myriad new classes of switched serial fabrics. Tools like Xilinx ISE Architecture Wizards manage these features for both single-ended and differential interface standards spanning a wide range of voltage, current and clock speeds.

Verification and optimization tools include propagation delay profilers, speed-driven placers and routers, power profilers, virtual scopes and waveform analyzers, pin out and area constraint editors, and bit-true simulators.

DSP programmers often create and validate algorithms and even entire signal processing systems on popular workstation tools like MATLAB. A comprehensive library of DSP, math, vector and matrix functions coupled with electronic models of popular hardware resources like mixers, oscillators and filters enable quick implementation of even the most complex applications.

During development, powerful signal generation and analysis tools allow designers to verify correct operation, validate dynamic range performance, and check for exception handling. Especially useful for signal processing, these tools operate equally well in the time or frequency domains with flexible presentation formats for easy viewing.

After successfully creating a DSP system, HDL code generation tools like SIMULINK can interface directly with the FPGA design tools.

Once in the FPGA development and verification environment, SIMULINK continues its role by providing hooks for stimulus test vectors and return paths for analysis of output signals. In this case, the signals are operating on simulation models of actual FPGA structures. MATLAB and SIMULINK are both products of MathWorks, and are well integrated through compatible links to Xilinx and Altera FPGA design environments.

This allows a broad class of FPGA users to take advantage of the extensive testing, documentation, standards compliance, speed and efficiency that might otherwise be unattainable with reasonable effort. The use of IP cores is a very cost-effective strategy to significantly slash FPGA development cycles for new technology products.

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.