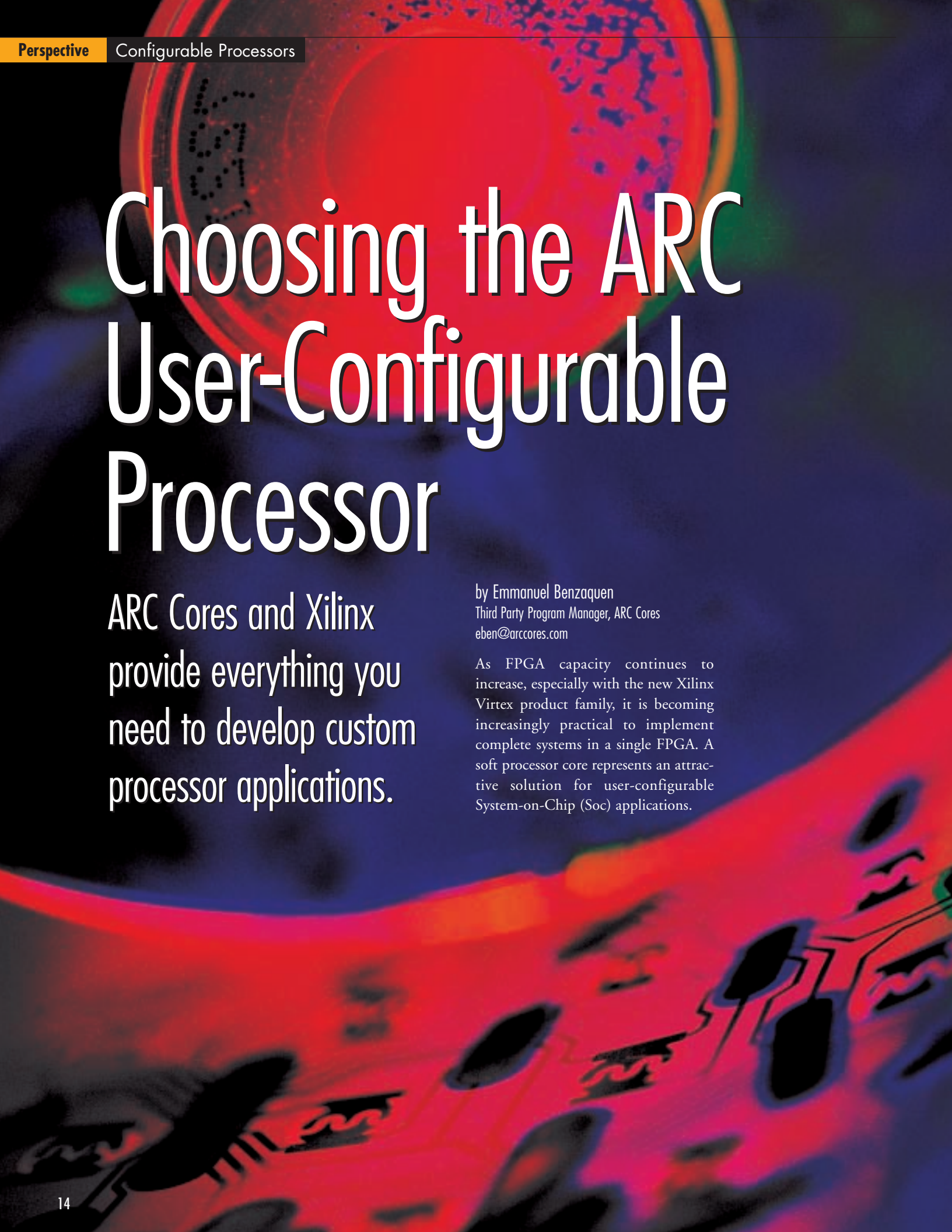


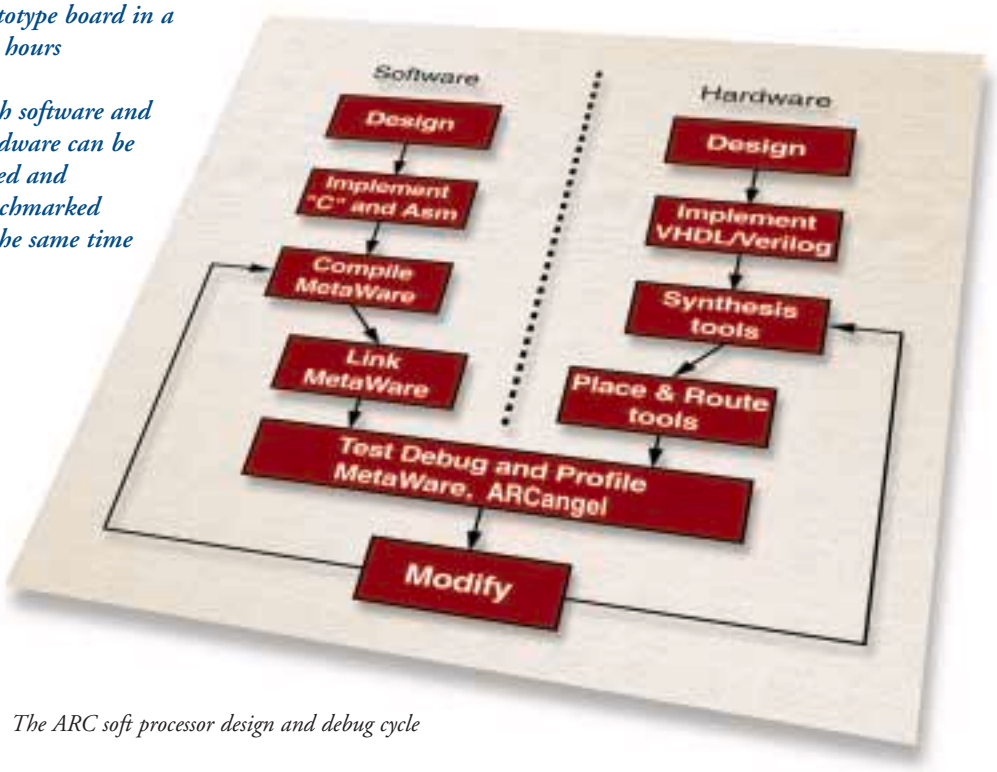
Choosing the ARC User-Configurable Processor

ARC Cores and Xilinx provide everything you need to develop custom processor applications.

by Emmanuel Benzaquen
Third Party Program Manager, ARC Cores
eben@arccores.com

As FPGA capacity continues to increase, especially with the new Xilinx Virtex product family, it is becoming increasingly practical to implement complete systems in a single FPGA. A soft processor core represents an attractive solution for user-configurable System-on-Chip (Soc) applications.

- *An ARC design can be turned from VHDL or Verilog into a configuration that runs on the Xilinx FPGA-based ARCangel prototype board in a few hours*
- *Both software and hardware can be tested and benchmarked at the same time*



The ARC soft processor design and debug cycle

Processor Cores Complement Programmable Logic

Traditionally, an important motivation for adding microprocessors into a design has been that software programmable solutions are easy to change and upgrade. Since FPGAs are, by definition, programmable, you can always upgrade them. System designers know that it is much easier to design and implement certain parts of a system using software, while hardware implementations offer greater performance. For example, you may want to take advantage of a large amount of low-cost software intellectual property (IP) that is available in C or C++ code, for functions such as protocol stacks and modem algorithms. You may also want to implement high-speed co-processor functions in hardware. You can get the best of both worlds when you combine the hardware re-programmability of FPGAs with the software programmability of microprocessors.

The Hard Versus Soft Option

Major FPGA vendors typically provide two different approaches to including processor cores in an FPGA. One approach offers a soft processor core that is provided in a synthesizable HDL format. This processor core is then included in a generic FPGA using the same design process as the rest of the logic. The second approach embeds a specific hard processor core (such as the PowerPC) into the FPGA. The most appropriate choice will depend on the application.

As a general rule, a hard processor core will offer higher clock speeds than a soft core. However, since the hard processor solution will require a specialized FPGA with dedicated processor buses and routing, it will be less flexible than incorporating a soft processor in a generic FPGA. In addition to performance and flexibility trade offs, the choice between a hard or a soft processor will also be influenced by the software applications you wish to run.

Since soft processors are available as synthesizable HDL, they inherently provide more design flexibility than hard processors because you can modify the core interface to fit better into a specific design. Some soft processors provide even greater flexibility by being configurable. A configurable processor core may include a graphical tool that enables certain functions to be included or excluded without having to manually modify HDL source code. As a result, you can create a processor core that is customized for your specific application by using the GUI.

The ARC processor is a perfect example of a soft and user-configurable core available for immediate use in Xilinx FPGAs.

Software Tools

The most important factor that will influence your choice of a soft processor is the software tool set that supports the code that will run on it. SoC designs can have lines of software code running anywhere from 1Kbyte to multi-megabytes. For applications that have only a few Kbytes of code, basic software tools such as an assembler may be sufficient. However once the amount of code starts increasing and becoming more complex, it becomes essential to use a high-level language like C or C++.

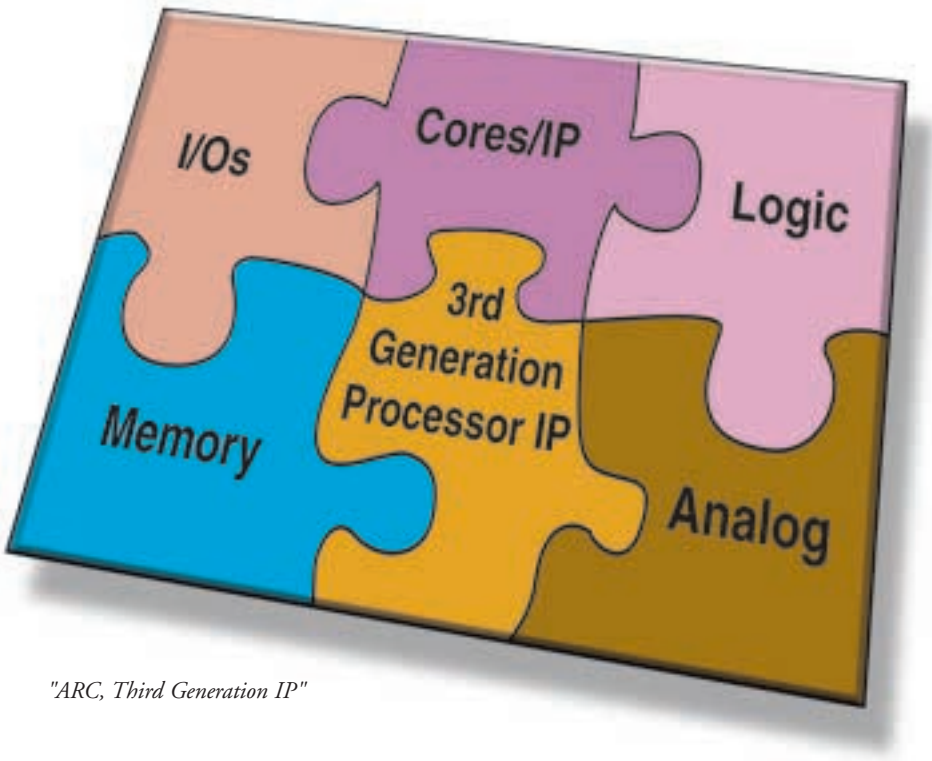
ARC Cores provide a complete set of high-level development tools customized for embedded applications, and offers both DSP (Digital Signal Processing) and general purpose control functions within the same processor architecture. There is no need to learn two different processor architectures and development tool environments.

Integrated Software Environment

Because of the typical complexity of the software code, ARC offers the Metaware development environment. This professional set of software development tools includes a C/C++ compiler, assembler/linker, and the SeeCode™ source-level debugger. Most importantly, it offers you the ability to debug the embedded software running on the processor in the FPGA. It is critical that the core and its host interface include execution control capabilities like breakpoint checking so you can break the program execution or monitor reads and writes to program variables.

As the software content of a design increases, another important factor is the range of applications supported and the available systems software. For example, if a design requires several hundred Kbytes of code along with standard communications software, such as TCP/IP protocols, you can save several months or more of design time by purchasing a real-time operating system (RTOS) that includes prepackaged protocols. ARC supports a large variety of commercially distributed RTOS from leading vendors and is constantly increasing their ease of integration.

In addition to the software tools and applications described above, another critical factor in choosing the ARC core is its level of flexibility. Unlike other configurable processors available today, which sometimes require you to manually "hack" the HDL code, the ARC processor core enables you to easily select special options for configuring the processor. Hacking the HDL code after configuring the processor core might break the core, or even make it incompatible with the software development tools. ARC provides the flexible ARChitect Graphic User Interface (GUI) that can be used to safely create your custom configured processor. This is very helpful when using a soft processor in an FPGA, and it allows you to experiment with different options and configurations within minutes.



"ARC, Third Generation IP"

- *The ARC IP is deeply embedded with the rest of the logic and interface directly with other customer logic function in the Xilinx FPGA*
- *"Gate-hungry" complex system buses and associated logic are no longer needed to reach high-performance because of the tight integration*

Instruction Set Flexibility

The instruction set is one of the most important aspects to consider when choosing a configurable processor. One potential disadvantage of soft processors is that they cannot attain the high clock speeds of a hard processor. For a conventional processor design, the clock speed is essentially the key determinant of performance. The ARC processor changes this equation by offering a configurable instruction set and the ability to add custom instructions. This enables you to accelerate an algorithm by selecting or adding a few appropriate (but powerful) instructions specifically needed for the application that is being executed. Thus, you can get the best of both RISC (Reduced Instruction Set Computer) and CISC (Complex Instruction Set Computer) processor design architectures. This approach provides high performance at lower clock speeds, while still maintaining a software programmable solution.

Instruction extensions are available from ARC and some third parties. Plug-ins can be used and implemented directly in the design. For additional capability, you can also create your own specific instructions. Custom instruction extensions offer you a particularly powerful way to accelerate application performance while retaining programmability. Consider the example of a DES (Digital Encryption Standard) encryption application: by adding specialist bit-permutation, cipher instructions and additional registers to hold the keys, it is possible to greatly accelerate a range of encryption algorithms.

To provide a truly configurable instruction set, it is also important that the number of clock cycles for an instruction extension is configurable. For example, the ARC processor enables the addition of multicycle instructions to the pipeline where desired, and single-cycle operations to proceed in parallel with long latency ones. This is an advantage over architectures that enforce a strict RISC paradigm where

every instruction must execute in a single cycle. Such restrictions may make it impossible to add very powerful, complex instructions that require multiple cycles to execute.

Interaction with Other Logic Functions

The ARC processor can further improve performance by enabling tight integration between the processor core and other logic on the FPGA. Traditional processor cores typically communicate with peripheral hardware via a system bus. To send data to the processor, the peripheral interrupts the processor, which then processes the interrupt using a software routine known as an ISR (Interrupt Service Routine). In addition to supporting this approach, ARC processor enables you to add new core extension registers. If desired, the new registers can be directly accessed by peripheral logic, enabling such devices to communicate with the processor directly. These alternative approaches can improve performance and reduce gate count by eliminating the need to duplicate a complex system bus and its arbitration logic in an FPGA.

It is no longer necessary to pass data via a bus or to interrupt the processor to have it load data from a memory-mapped register. Since the special registers are unique to a particular piece of peripheral logic, there is no need for any decoding or arbitration logic. The firmware simply selects the special purpose registers to communicate with the peripheral.

In addition to providing extension registers, configurable processors like the ARC core can also simplify integration with additional logic by providing multiple buses. This approach enables operations residing on separate buses, such as instruction fetches, load/stores, and communication with peripheral logic. As a result, the bus protocols of each bus can be relatively simple since there is no need to arbitrate between multiple devices attempting to control one bus. The ARC processor has four buses, consisting of instruction and data buses (Harvard architecture), a bus directly into the processor registers (primarily used for debugging), and an auxiliary bus (typically used to connect peripheral logic). The auxiliary bus has a very simple interface that virtually enables peripherals to be connected with just a few wires. This is well suited to FPGAs where there is no actual bus, allowing peripherals to be efficiently connected in a point-to-point manner.

Tool Configurability

Any processor that offers a high degree of configurability must also offer equally configurable software tools and a debugging environment that work in coordination. It is of no use to add new instructions to the processor if there is no way of telling the compiler and assembler about them so that actual software programs can take advantage of them. In a similar vein, the compiler must let you specify which instructions will be present in the processor, as well as be able to take advantage of features such as multipliers or barrel shifters when they are included. In fact, software tool configurability is one of the greatest challenge in providing a truly configurable processor solution.

ARC and Xilinx are responding to this challenge by offering a complete "plug and play" solution to FPGA designers. In addition, the ARC tools suite allows you to enhance the original configurations offered in a simple manner.

Conclusion

Soft processor cores give you the ability to include processors in standard FPGAs. Configurable cores can help you achieve higher performance at lower clock rates through instruction extension and peripheral logic integration. ARC and Xilinx offer the perfect combination of a configurable core with powerful extensions and third party "plug-ins," in addition to a complete development environment and operating system support, ready to use with Xilinx FPGA technology.