Configurable Processors Offer More Ways to Improve System Cost and Performance

Processors with configurable architectures and instruction sets, along with a new generation of tools, offer developers new opportunities to fine-tune system bottlenecks.

by Stefano Zammattio
ARC International

Over the last few years, the number of vendors offering configurable processors has grown considerably. Although the benefits of configuring peripheral intellectual property (IP) modules are clear, is the same true of configuring the processor, the very heart of the embedded system?

With fixed processors, developers attempting to hit performance and cost targets are limited to boosting performance by simply changing processors or increasing the clock frequency. These tend to be poor solutions because they usually involve significant increases in power consumption, cost and difficulty implementing the final silicon. With configurable processors, developers have more options, since they can tailor processor performance or system configuration to meet the needs of the application by modifying the IP.

Development tools have also matured, allowing more in-depth analyses of system and processor performance and thus a better means of identifying the causes of performance problems. A configurable processor lets designers directly address specific performance issues, whereas with a fixed processor they are limited to designing around them. Consequently, many developers are beginning to try out this new technology. Those who have already tested it often discover that real performance gains are possible.

Performance: A Multi-Stage Pipe

The performance of an embedded system can be visualized as a multi-stage pipe, where each stage can restrict the application for a specific reason, sometimes leading to unexpected limitations in system performance (Figure 1). For

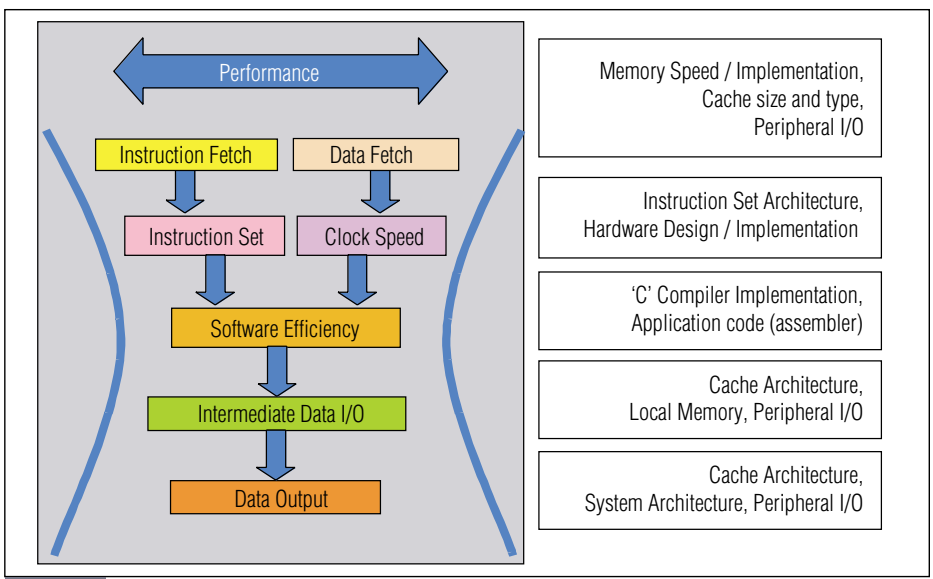

Figure 1 An embedded system can be portrayed as a multi-stage pipe where each stage can limit the system's overall performance.

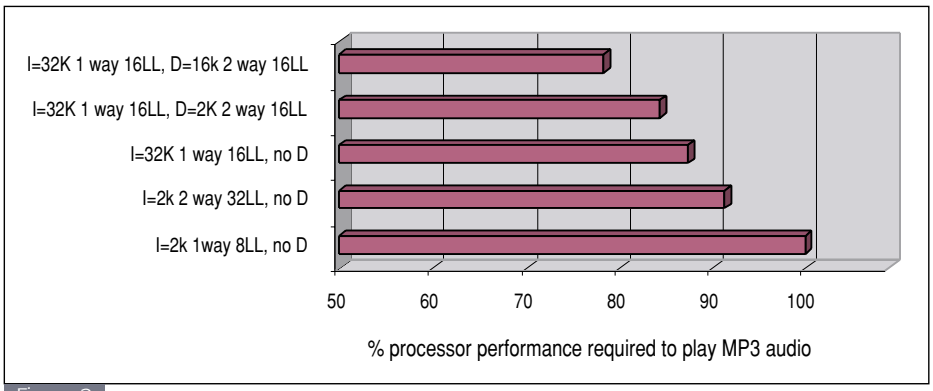

Figure 2 Decoding MP3 with different processor cache configurations can result in significant variation in processing power requirements.

example, a combination of the memory fetch latency, cache size and type, and peripheral I/O implementation can place a heavy load on the system/memory bus, limiting the performance of the processor, and therefore of the system.

Developers commonly estimate performance load on the system/memory bus by simply adding peripheral bandwidth to that required by processor data and instruction fetches. This would seem to be a straightforward calculation. However, the required processor bandwidth is, in fact, not easy to determine because cache behavior is difficult to predict. Such prediction is especially difficult if most of the application code has been developed using a compiler. This problem can be exacerbated by the code's heavy use of functions and intermediate I/O, such as stack operations and the storage of global and intermediate data values in memory. The combination of a slow clock speed and an instruction set that is relatively ineffective in terms of the needs of the application will result in a system that is limited by the processing power of the CPU.

The system's software efficiency can also be hampered by an ineffective compiler. This is not always an obvious issue, since effectiveness varies tremendously with the application, coding style and processor architecture. Coding in assembler is an easy solution to this problem, but then the limitation becomes the programmer's ability to develop fast, robust code in the time available.

In addition to these factors, an embedded system may have real-time I/O or processing requirements. Whether these are met through the use of an RTOS or not, there will always be a certain amount of overhead involved in stopping and restarting the system to service the real-time requirement.

Performance Increases with Configurable Processors

The primary benefit of configurable processors is that the "width" of the performance pipe can be changed at any stage. This flexibility lets developers tweak their systems to alleviate system bottlenecks and deliver much higher performance than is normally obtainable using a fixed

processor. Two different examples will be discussed here: improving the performance of a well-constrained application (an MP3 decoder) by varying cache parameters; and optimizing performance of an application that relies heavily on a single algorithm (the Data Encryption Standard, or DES) by the generation of a custom instruction.

The goal of the first project was to reduce system cost by minimizing cache size without affecting performance. This was achieved by evaluating system performance with a range of different cache configurations.

The code was written primarily in assembler, and had a relatively straightforward data I/O requirement. One would therefore expect that the cache configuration would make little difference, as long as the caches were large enough. But, as shown in Figure 2, even in this apparently simple case, cache configuration can make a difference of more than 20% in performance. Profiling an MP3 decoder takes many processor cycles, so a cycle-accurate model of the processor was used to generate the data. Results were also confirmed by running the same application in an FPGA implementation.

In the case of a system coded in highly structured C, with a much larger number of stack and global operations, it becomes very difficult to predict the cache's optimal configuration. With a fixed processor, the developer's choices about cache sizes are limited. However, with a configurable processor there is a full range of choices, including no cache at all. Thus, the developer gains control over cache configuration, as well as having additional alternatives for changing the processor's I/O characteristics. For example, the ARCtangent has a second, separately addressable region known as auxiliary space, which can be used to alleviate the load on bandwidth by diverting data flow away from the system bus. In an MP3 player, the data output can be streamed to auxiliary registers, hence removing a 44-kHz memory bandwidth requirement from the system bus.

Demonstrating that the addition of custom instructions to a configurable processor can greatly improve its application-specific performance was the goal of the second project. The DES algorithm, used heavily in network applications, frequently repeats a function known as the S-box. This function is awkward to implement with the standard logical and numerical operations of fixed processor instruction sets, but is well suited to hardware acceleration.

The effect of implementing the S-box in a custom instruction was a reduction of the system load required for Internet Security Protocol (IPSec) encryption and decryption. Such optimization could either free up resources for other tasks without increasing clock frequency, or allow the processor to run at a lower clock frequency in order to simplify system design, cut costs without sacrificing performance and reduce power consumption.

A DES acceleration instruction incorporating the S-box function was built and integrated into a configurable processor. IPSec protocol code and software development and analysis tools were then used to compare the performance of the processor before and after customization. The reduction in the number of processor cycles required for the S-box function resulted in

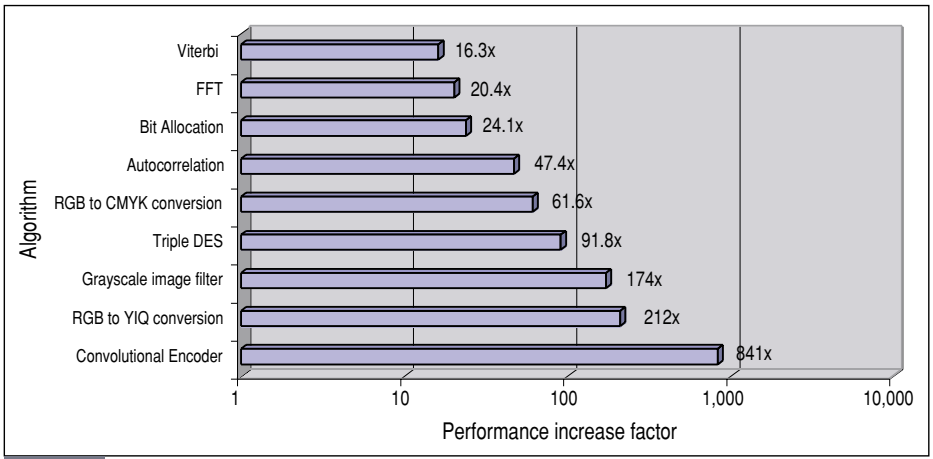

Figure 3 The performance of a configurable processor can be greatly increased by implementing custom instructions.

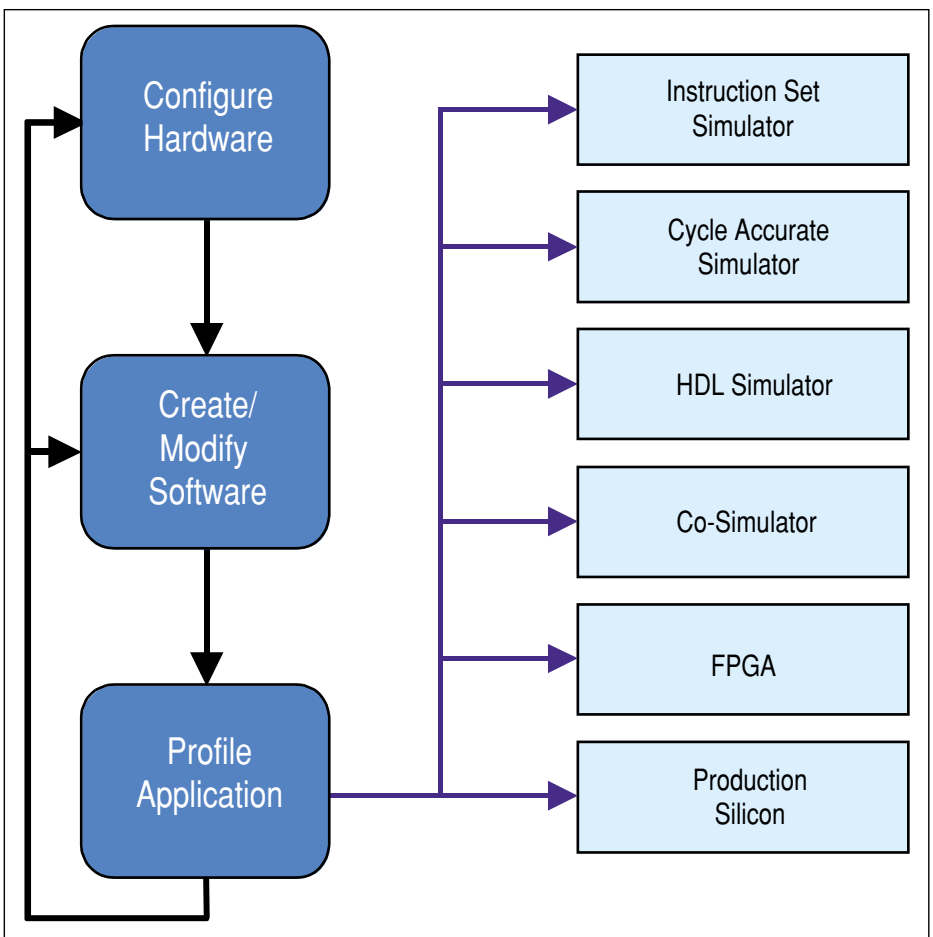

Figure 4 During design, developers can use a range of different tools to profile system performance.

an overall performance increase of 91.8x for Triple DES (Figure 3).

Most algorithms can be accelerated in a similar manner by the implementation of custom instructions. Figure 3 also shows other examples of code acceleration using custom instructions. The algorithms shown are those used by the Embedded Microprocessor Benchmarking Consortium (EEMBC) benchmarks suite.

A secondary, but important, benefit of such optimization is that the code required for the accelerated function is often reduced, since much of it is replaced by a single instruction. For example, the DES algorithm code size was reduced by 63%, from 6.5 K to 2.5 K. Reduced code size can lead to significant cost savings for the developer.

Naturally, all of these modifications add extra work to the development and verification phases of the project. In each project there will be unique tradeoffs between working around the limitations of a fixed processor versus the effort involved in customization. However, many developers aren't yet able to assess the benefits of one approach over the other, since the use of configurable processors is still relatively new, and because such judgments can only be done on a case-by-case basis by project designers with experience in both methodologies.

As developers become increasingly aware that it is system performance—not just processor performance—which matters, they are demanding more powerful tools and techniques for profiling the systems they design. EDA tool and IP vendors are addressing these demands in several ways. Cycle-accurate C models of many IP modules are now available so that designers can simulate their systems entirely in software. In addition, HDL simulators are increasing in power, allowing developers to simulate the actual hardware they have designed. These simulators also have open interfaces so that software debuggers can be connected directly to the HDL system simulation.

These same open interfaces are also used by co-simulation tools to allow HDL simulators to interact with other systems, for example, a cycle-accurate C model. This produces a hybrid system model in which large complex blocks, such as processors, are modeled in C while the hardware being designed and its system configuration are modeled in HDL simulation. The benefit of this is that developers can leverage the speed of the cycle-accurate C model while retaining the accuracy and visibility of the HDL simulation. Without this it would simply not be possible to debug large hardware/software systems in a reasonable amount of time.

A range of different tool technologies is now available that lets developers profile their system's performance and gather much more information about performance limitations long before the manufacturing phase (Figure 4). Each tool has a specific place in the development process.

At the very beginning of the design flow, for example, the instruction set simulator will be used. Later on, a co-simulation tool may be more suitable. Being able to quantify performance limitations as early as possible in the product design cycle avoids the disaster of having to go back to the drawing board, especially after first silicon returns from manufacturing.

Instead, this information can be used to tailor the design during product development to ensure that performance targets will be met. With a fixed processor, this can be extremely difficult to achieve. Since the processor is a fixed unit, a major restructuring of the rest of the system and/or the software may be required. Simply increasing the clock frequency is a poor solution, because it increases power consumption and makes the final silicon implementation more difficult. Developers have more options for solving these problems when the processor performance or system configuration can be modified to meet the needs of the product.

It is clear that significant benefits can be obtained from the use of configurable processors, but developers need to become more familiar with the types of potential solutions that these processors can offer and the development effort required to implement them. Clearly, judgment calls must be made in weighing the tradeoffs of the amount of effort involved to work around fixed processor core issues against the amount of effort required to customize the processor. Such tradeoffs are highly specific to the application and system design. As with all new technologies, it will take time for developers to fully understand the implications of processor configurability so that they can leverage the technology to their best advantage.

One of the main reasons that developers still choose fixed IP is legacy. However, as system requirements continue to outgrow the capabilities of the fixed processor implementations on which they are based, it seems natural for developers to choose a configurable processor technology, even if they don't intend to take advantage of its configurability in the first version of the product. Using the processor as a fixed core in the first product generation can give them time to learn about and understand the technology. Then, at some point during that first version, or in a subsequent product revision, developers can modify the processor and/or the system to give the boost in performance that is usually required for the product's next generation, without changing the processor and tool chain involved. ◢

ARC International
Elstree, UK.
(+44) 208-236-2800.
[www.arc.com].

FPGAs Enable New Era of System Design

Once relegated to prototyping duties, FPGAs are changing the landscape of system-level chip design.

by Clay Johnson
Xillinx

After years of playing the essential "early ASIC-prototype" role, field-programmable gate arrays (FPGAs) are ushering in a new era of system design. Today's high-density FPGAs have evolved to encompass the key requirements to be the optimal technology for System-on-Chip (SoC) product applications. FPGAs can also be called Programmable Logic Devices or PLDs, however when and where a device is programmed is an important cost and functional distinction in defining this new era of design.

Market pressures continue to force evolution in design methodology. Such pressures make it essential to integrate all of a products' digital electronics into a single SoC. The SoC design methodology, and its implied IP modularity, has become the preferred industry design standard. The popularity of SoC methodology is primarily due to increased design complexity that pushes the average design to 30 million gates and beyond.

SoCs Ramp-Up Capabilities

SoCs designed using field-programmable technology offer an advantage in meeting rapid time-to-market and time-in-market requirements, already displacing standard-cell ASICs and traditional MOS gate arrays in telecommunications and networking applications. FPGA and PLD manufacturers continue to push the system-level integration envelope. They're doing so in terms of performance—raw clock speed and hard macros, in terms of advanced connectivity with high-speed serial I/Os), and lower costs. They're also easing the design process by integrating push-button intellectual property (IP) components (Figure 1). This combination of features helps further minimize early product development risk and better meets aggressive product introduction schedules—key advantages in today's changing markets.

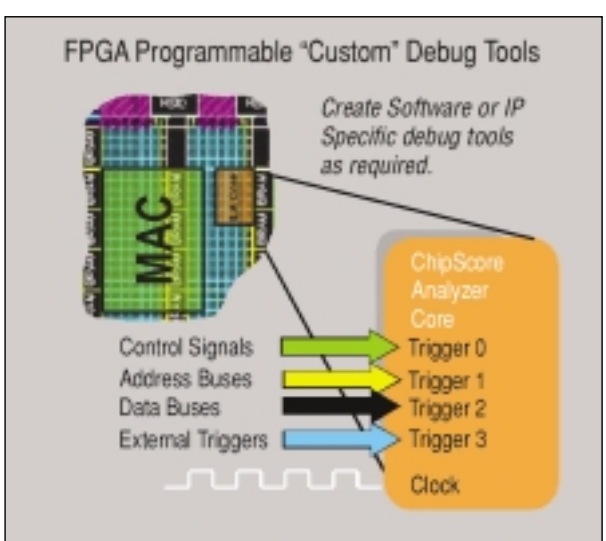

Figure 1 On-chip resources like Xilinx's ChipScope Analyzer core let users create software- or IP-specific debug tools as required.

The rapidly developing trend favoring programmable logic has not escaped the notice of traditional pure-play ASIC manufacturers. LSI Logic was one of the first ASIC manufacturers to attempt to add programmable logic to its ASIC technology. LSI assisted in technology development with a now out-of-business startup called Adaptive Silicon. LSI lost interest after it became apparent that its demonstration product implemented in its 0.18-micron technology didn't generate money-paying interest.

Other ASIC vendors are undeterred by past failures and are increasing efforts to add some level of programmability to their ASIC products. NEC announced its ISSP Modular Array ASIC, and Hitachi seems to be on track with Triscend-based SoC development. More recently, the two leaders in their respective fields, IBM and Xilinx, solidified its long-term partnership by announcing its XBlue FGPA fabric for IBM's ASIC-based products. Yet another new partnership was announced by Infineon and Actel to further develop Actel's ProASIC FPGA technology in Infineon's next-generation sub-100 nanometer silicon technology.

Proven Role For Prototyping

Most ASIC engineers are experienced with using FPGAs in prototype platforms. It's the only way to successfully start the time-consuming system software develop-

Manufacturer Family Device	System Gates	Total RAM (bits)	Clock Freq (MHz)	Flip Flops or LEs	Prog. I/Os	Other Blocks DSP, PLL	Process Tech (m)	Metal Layers	Voltage
Xilinx Virtex-II Pro 2VP1125	10 M	10 M	420	125,136 LCs plus 4x PPC cores	1,200 + 24x 3.125 Mbits/s	28 DSP Blks (556 18x18 Mults)	0.13	8	1.5 V
Xilinx Virtex II XC2V10000	10 M	3.5 M	420	122,880	1,108		0.18	8	1.5 V
Altera Stratix EP1S120-(2H03) EP1S80-(3Q02)	10 M+	Up to 10 M	-	114,140 LEs	1,314 user I/Os 116 LVDS links	28 DSP Blocks (224 9x9 Mults)	0.13	8	1.5 V
Altera Excalibur EXP2A90	7 M	1.5 M + ARM 922T core	-	-	1,140		0.15	8	1.5 V
Altera APEX II EP2A90	7 M	1.5 M	-	-	1,140		0.15	8	1.5 V
Lattice ispXPGA 1200	1.25 M	660 K	-	15,376 LUTs	496 user IO + 20 pr SerDes	On-chip E^2 cells	0.15	8	1.5 V
Lattice ORCA Series 4	899 K	552 K 148 Kb Embedded	-	16,192 LUTs	466		0.15	8	1.5 V
Actel Axcelerator AX2000 Q402	Up to 2 M	339 K	500 MHz	32 K Antifuse modules Live at power-up	684 user IO	64 bit per pin FIFO	0.15	7 L	1.5 V
Actel ProASIC+ APA 1000	1 M	198 K	-	12 M flash cells	712		0.15	8	1.5 V
Quicklogic Eclipse QL6600 QL901M (MIPS)	583 K	82,900	250	9,600	512		0.25	5	2.5 V

Table 1 The five leading FPGA vendors are Xilinx, Altera, Lattice, Actel and Quicklogic. The table lists the features and capabilities of these vendors' newest and most promising programmable logic.

ment process. Frequently, software development is the critical path, consuming the majority of resources—another big reason why modular SoC design methodology has become an industry standard (Figure 2).

Large networking and telecommunication OEMs are prolific users of programmable logic. OEM designers use FPGAs to overcome the impact of rapidly changing requirements and standards. Experienced designers, who have participated in successful design projects, know there isn't a better design "risk-management plan" than having key components implemented in FPGAs. The use of programmable logic makes it easy to track changing standards, fix bugs and optimize performance.

Silicon manufacturers and OEMs look toward a more flexible SoC design to improve their time–to-market and time-in-market. Flexible, effective use and reuse of IP seems to be the new mantra and the best way to close the design-productivity gap that results from increasingly larger and more complex designs.

Lowering cost, or lowering the total cost-of-ownership is a recurring theme these days. At July's Metro Optical Networking Forum the cost-of-ownership issue and adaptability to changing standards were emphasized in many presentations and keynote speeches. The experts agreed that the industry had to cooperate more on reducing time-to-market and developing systems that would rapidly adapt to change.

The selective use of FPGAs or FPGA fabric will enable the dynamic reconfiguration of IP for rapid optimization to a particular application. The in-field programming aspect of FPGAs is the major attraction in using these devices. Modular Array ASICs that are programmed during device fabrication or one-time-programmable (OTP) PLDs do not significantly lower total cost-of-ownership—entire boards or systems would need to be replaced in order to upgrade hardware or add services.

Adaptable Computer Systems: Holy Grail

The entertainment industry provides countless fictional examples of self-healing or self-building computer systems as the ultimate implementation of Artificial Intelligence (AI). Fully adaptable computer systems are actually a reality today empowered by the hardware and software programmability possible in today's processor-equipped FPGA devices. A good example of "adaptable systems" is the push within the Metro Optical Networking market to create multiservice provisioning platforms (MSPP)—a single MSPP is easily configured to integrate new value-added services. Hybrid FPGA and ASIC devices enable the creation of just-in-time adaptable computer systems. Successful hybrid devices are either ASICs with FPGA fabric or FPGAs that implement specific functions using traditional ASIC technology.

Telco carriers demand equipment that's easy to configure for new added-value services. To serve such needs FPGA and ASIC technologies are likely to merge in next-generation chip products. The goal is to bring the high-integration capability and low-cost aspect of ASIC devices together with the ability to make late hardware design changes. Telco carriers and service providers have already successfully incorporated such field "software and hardware" upgrade capabilities into many systems. Lucent for example promotes its Smart Upgrade capability as a means to allow in-service software and hardware upgrades as a way to minimize service disruptions.

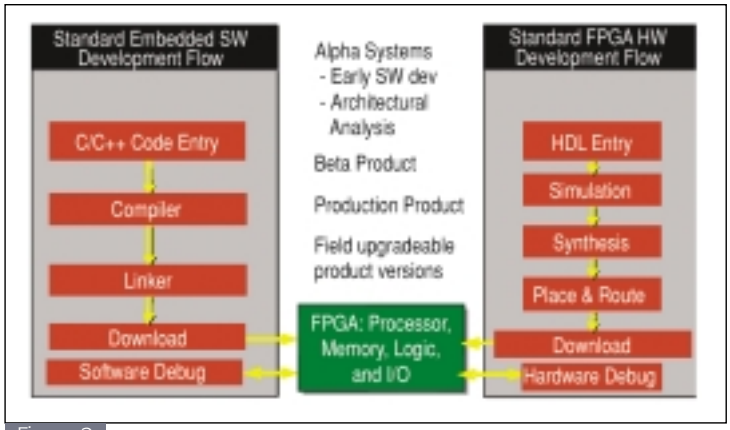

Figure 2 Software development is often the critical path, consuming the majority of resources. FPGAs make those efforts easier by supporting a highly iterative system design process.

Wafers Near a Cost/Yield Collision Point

The driving force behind FPGA's appeal is simple economics. The biggest economic impact results from a "one-two" punch: the move to 300 mm or 12-inch wafers and substantially higher mask costs of sub-100 nanometer silicon technology. The chip industry's move from 130 nanometer to 90 nanometer silicon technology has created a compelling need for ASIC and ASSP vendors to look at new methods to minimize risk and maximize cost-effective silicon fabrication.

A key future issue is risk management. The number of die at risk in a single fabrication run increases dramatically when 300

mm wafers are employed—about 2.25x the area of a 200 mm wafer-size and a 27 percent smaller individual die size from 130 nm to 90 nm. New cost-effective wafer fabrication requirements move cost-effective unit volume numbers even higher, and potentially beyond the range of all but the highest volume or fully proven applications.

Looking at FPGA technology as the solution to the cost/yield issue makes sense. A single ASSP, without programmable capability and with a fixed feature set, must have its total cost amortized on a per version basis—the fixed function ASSP must target a specific high-volume application for it to remain cost competitive. ASIC development costs increased dramatically with the introduction of 90-nanometer silicon technology—million-dollar mask sets—and is only part of the story. Many applications are highly integrated and more complex designs inherently have higher risk of design error. Late design cycle changes and schedule extensions will often make the difference in a product's ultimate success or failure.

The use of embedded core-based FPGA technology to implement unproven logic in a complex ASIC or ASSP design can in many cases guarantee shippable first silicon. It would also enable application-specific optimization that is essential for product differentiation of designs based on the same ASSP. Appling the right ASIC intellectual property to a large FPGA can provide the ultimate programmable and configurable device. This solution may meet performance requirements of larger and more complex designs at little or no increase in device cost.

FPGA and ASSP Examples

The five leading FPGA vendors, in order, are, Xilinx, Altera, Lattice, Actel and Quicklogic. Table 1 lists the features and capabilities of these vendors' newest and most promising programmable logic. The devices from the two leading programmable logic device or FPGA vendors, Xilinx and Altera, are most promising due to the larger number of devices per family. These vendors have new hybrid ASIC/FPGA devices that provide more than 10 million ASIC system gates composed of processor cores, memory, data path and DSP elements, programmable logic gates, and programmable connectivity. The prospect of reconfiguring processor software and entire processing fabrics in the field provides unprecedented application and product flexibility.

Current FPGA hybrid devices have implemented PowerPC, ARM and MIPS processor cores. The breadth of soft processors is growing with both Altera and Xilinx supporting multiple versions of MicroBlaze, PicoBlaze and Nios cores. The new FPGAs also can be used to embed DSP building blocks capable of configuring various width multipliers. Standard supported multipliers are 9 x 9, 18 x 18, and 36 x 36. These standard multiplier configurations deliver up to 800 billion multiply-accumulate operations (MACs) per second. That's more than enough to tackle digital video, embedded computing and broadband wireless applications. In comparison to this capability, the fastest DSP only reaches about 8.8 billion MACs per second.

Connectivity is King

It wasn't too long ago that FPGAs struggled to meet PCI signal timing requirements easily achieved using standard ASIC technology. FPGAs have taken the lead over ASICs in supporting new I/O standards like PCI Express and RapidIO. The I/O pin count of the largest FPGAs exceeds 1000 pins of user I/O that is highly programmable. Programmable User I/O in FPGAs supports over 20 state-of-the-art I/O standards such as PCI, PCI-X, PCI Express, Serial and Parallel RapidIO, HyperTransport and a number of low voltage differential signaling (LVDS) standards. Newer serial I/O standards provide unparalleled connectivity.

FPGA IP now spans three forms: soft, firm and hard macros. An FPGA soft macro is a synthesizable HDL description of a functional component that can usually target FPGAs, ASICs, or MOS GAs. Previously, HDL components optimized for FPGAs were incompatible with ASIC design flows. Now, HDL coded for ASICs may need only a few minor "tweaks" to be FPGA-optimized. The advantage of soft cores is that their HDL code is "technology independent", meaning it can target just about any FPGA or ASIC design flow.

For its part Altera allows its Nios soft-processor core to be retargeted using ASIC design flows. Nios is a RISC-based configurable and scaleable processor that offers a 16-bit instruction set and 16-/32-bit datapath. The newest version of the Nios core can be implemented on an Altera APEX 20K or Stratix device. An example of a firm macro is an HDL design file that implements a specific function, like a RapidIO interface, in the FPGA or programmable fabric. Examples of hard macros are the hard 405-based PowerPC processor core implemented in Xilinx's Virtex-II Pro family or the hard 922T-based ARM core implemented in Altera's Excalibur family.

Actel
Sunnyvale, CA.
(408) 739-1010.
[www.actel.com].

Altera
San Jose, CA.
(408)544-7000.
[www.altera.com].

Lattice Semiconductor
Hillsboro, OR.
(503) 268-8000.
[www.latticesemi.com].

QuickLogic
Sunnyvale, CA.
(408) 990-4000.
[www.quicklogic.com].

Xilinx
San Jose, CA.
(408) 559-7778.
[www.xilinx.com].