Closing the SoC Design Gap

Jörg Henkel, NEC Laboratories, Princeton

System-level design and extensible processors can bridge the gap between silicon technology and actual SoC complexities.

Embedded systems are ubiquitous, and they represent the major market for microprocessors. In fact, more than one billion microprocessors find their way into embedded systems every year, and this number will continue to grow rapidly.

One reason for this phenomenon is that intense competition among manufacturers—especially for mainstream consumer products—has shortened the life cycles for products that use embedded systems. For example, cell phone manufacturers now release two major new product lines each year compared with only one just a few years ago.

At the same time, product functionality and hence the complexity of underlying embedded systems are rapidly increasing. Cell phones, for instance, now provide many functions beyond their core functionality, such as Web browsing capabilities, personal digital assistant functions, short message services, and even gaming.

DESIGN CRISIS

Shorter product life cycles and increased functionality create a demand that matches well with the International Technology Roadmap for Semiconductors' prediction of 1 billion transistors on a single chip within this decade. However, current mainstream system-on-chip (SoC) designs do not yet fully exploit the 100 million transistors per chip possible with today's mainstream silicon technology.

Electronic system-level (ESL) design tools offer a way of addressing this "crisis of complexity," as Gartner Dataquest's Gary Smith described it at this year's Design Automation Conference. Without extensive use of ESL design tools, though, it is predicted that by 2005, when 250 million gates per SoC become feasible, most SoCs would use only around 50 million.

The increasing gap between silicon technology and actual SoC design complexities is one indicator of the need for ESL tools combined with intense use of off-the-shelf components. Another is a decrease in the number of design starts for application-specific integrated circuits relative to the number for application-specific standard products—that is, customized ICs sold to more than one customer. In fact, ASSP design starts are currently outpacing ASIC design starts by about 5,000 to 4,000, respectively (B. Lewis and S. Hsu, *ASIC/SOC: Rebuilding after the Storm,* market report, Gartner, 2002). The increasingly high nonrecurring engineering (NRE) costs of migrating to 130-nanometer ASIC technologies only amortize for very large volumes—perhaps several million units, depending on the design.

SOC DESIGN TRENDS

Early embedded SoCs were of low complexity, typically comprising one or two programmable processing units—say, a microcontroller and a digital signal processor—plus some custom hardware, peripherals, and memory. Current SoC designs, however, use a rapidly increasing number of processing units. Further, programmable processing units are replacing complex custom logic blocks, thereby reducing SoC development times and NRE costs while offering comparable performance.

Certainly, custom logic blocks will continue to address such embedded SoC requirements as very high performance and ultra low power consumption. But their complexity will shrink. Recent in-house designs show that even subblocks will migrate to low-complexity processing units with 10k-20k gates and a basic instruction set. Designers can use this migration to trade performance, power, and other requirements for flexibility whenever possible.

This dual trend—increasing the number of programmable processing units on SoCs while decreasing the overall gate count for custom logic—contributes to the declining ASIC design starts in favor of ASSPs. SoCs for embedded systems designed with both trends in mind already feature 10-15 programmable processing units on a single chip. The trend is expected to continue, yielding a "sea of processors"—

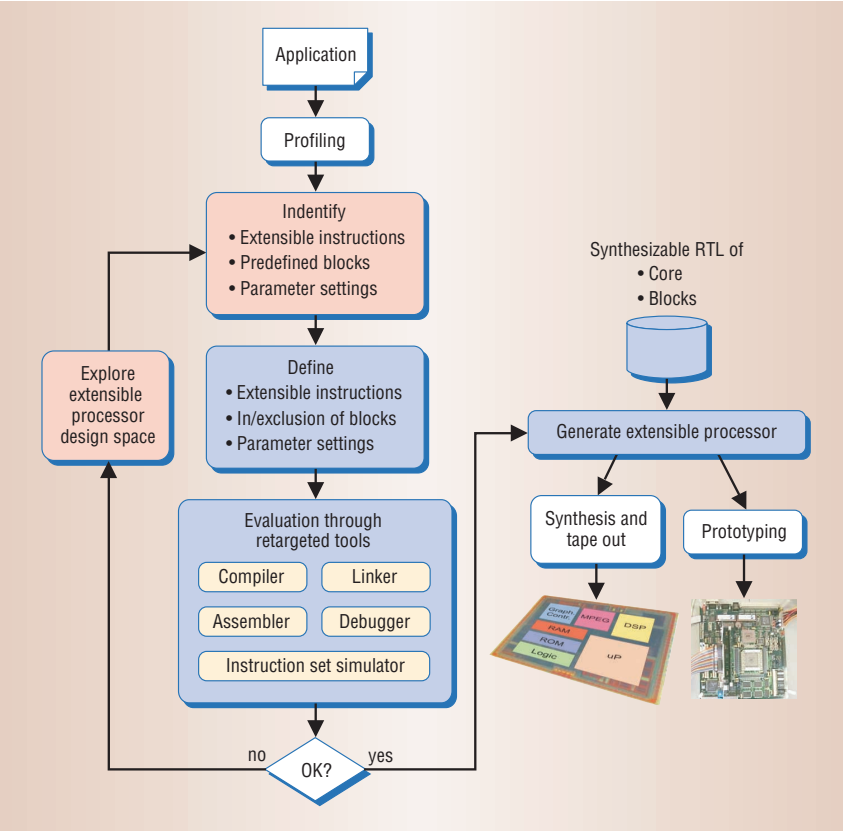

Figure 1. A simplified generic design flow of an extensible processor platform.

hundreds of heterogeneous processors on SoCs connected by a network-on-chip (NoC)—by the end of the decade.

EXTENSIBLE PROCESSOR PLATFORMS

Application-specific instruction-set processors (ASIPs), which belong to the ASSP circuitry group, offer a possible means of overcoming the crisis of complexity in SoC design. Extensible processor platforms represent the state-of-the-art in ASIP technology. Their customization typically addresses three architectural levels that vary depending on the platform vendor:

- *Instruction extension.* The designer can define customized instructions by specifying their functionality. The extensible processor platform will then generate the extended instructions that then coexist with the base instruction set.
- *Inclusion/exclusion of predefined blocks.* The designers can choose to include or exclude predefined blocks as part of the extensible processor platform. Block examples include special function registers, built-in self-test, multiply-and-accumulate operation blocks, and caches.
- *Parameterization.* The designer can fix extensible processor parameters such as instruction and data cache sizes, the number of registers, and so on.

Commercial platforms for extensible processors are currently maturing in large SoC design projects such as NEC's TCP/IP offload engine, which integrates 10 extensible processors on a single chip. Major extensible processor platform vendors include Tensilica, Improv Systems, ARC, Coware, and Target Compiler Technologies.

Generic design flow

Figure 1 shows a generic design flow of an extensible processor platform. The goal is to customize the extensible processor to a specific application that will run on it. The application may be available in a language like C/C++ as an executable specification. The designer starts profiling the application using an instruction-set simulator (ISS) of the target processor. The profiling reveals computational bottlenecks for which possible instruction extensions, inclusions or exclusions of predefined blocks, or parameterizations might improve performance or power characteristics. This step is an art and requires significant design expertise.

After identifying a set of possible extensible instructions, the designer uses a custom language to define them—including, for example, an extensible instruction's functionality, pipeline scheduling, instruction word format, and so on.

To evaluate these customizations, designers can use *retargetable tool generation*. Retargetable techniques can automatically generate compilers and simulators that are aware of the new or extended instructions and other customizations and thus can generate or simulate code that is aware of the customizations. The designer uses the retargeted ISS, compiler, assembler, and other tools to evaluate whether the application can meet the performance and power constraints.

Designers can iterate this step often and fast to get early feedback during the *design space exploration*. Once the application (compiled with the extended instruction set) meets the design constraints, the platform uses the extensible instruction definitions and other customization levels to automatically generate the extensible processor's synthesizable RTL.

The process concludes with a regular synthesis flow down to tape out or an evaluation based on a rapid prototyping environment.

Open issues

Automatically determining the right set of extensible instructions for a given application and its constraints remains an open issue. The orange boxes in Figure 1 represent this step, which involves optimization and search through a large design space.

Another question is how to have many extensible SoC processors communicate with each other via a customized NoC design. Also, retargetable tool generation needs further research—for example, to enable a retargeted compiler to make full use of designer-defined extensible instructions during the optimization phase.

These and similar issues are currently hot research topics.

SHIFT IN SOC DESIGN DISTINCTION

An embedded SoC's custom logic hardware blocks are often considered the key to distinguishing vendor A's design from vendor B's. The distinguishing characteristics are mainly performance and power consumption.

As custom logic migrates to programmable processing units, the distinctions between designs are shifting—in part to the process of customizing extensible processors. Plus, because developers eventually must program extensible processors, more design expertise is directed to software.

The increasing size of embedded software bears witness to this trend. Recent SoC designs feature more than a million lines of embedded code, and embedded software is taking more engineering resources than hardware design (P. Magarshack and P. Paulin, "System-on-Chip Beyond the nanometer Wall," *Proc. 40th IEEE/ACM Design Automation Conf.*, ACM Press, 2003; pp.419-424).

ASIC technologies, on the other hand, have ramped up against the trend of very high NRE costs associated with the migration to 130-nm processes and beyond. Recently, *structured ASICs* have emerged, which have most of the metal layers predefined in the form of prebuilt logic blocks, configurable memory blocks, and test structures. Customization takes place only in the upper two or three metal layers. Hence, NRE is far lower, and even unit costs decrease because all customers share the same prefabricated die.

Structured ASICs are economical at manufacturing volumes of less than 100,000, and some vendors are already using them in the newest 90-nm technology.

Advances in system-level design methodologies and an increasing migration of custom logic into programmable processing units in the form of, for example, extensible processors may well bridge the gap between available silicon technology and actual SoC complexities. However, SoCs comprising 1,000 processors at a billion transistors by the end of the decade will require research advances in key areas like ESL design methodologies and NoC architectures. ■

Jörg Henkel is a senior research staff member at NEC Laboratories, Princeton. Contact him at henkel@nec-labs.com.

Editor: Wayne Wolf, Dept. of Electrical Engineering, Princeton University, Princeton, NJ, 08544-5263; wolf@princeton.edu