For Programmers, Multicore Chips Mean Multiple Challenges
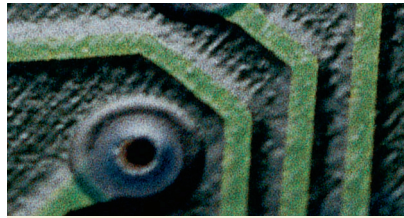
David Geer

Manufacturers have found themselves unable to effectively continue improving microprocessor performance the old-fashioned way—by shrinking transistors and packing more of them onto single-core chips.

This approach generates too much heat and uses too much power, noted Louisiana State University (LSU) professor Thomas Sterling.

In response, vendors are increasing performance by building chips with multiple cores. Principal microprocessor manufacturers Advanced Micro Devices (AMD) and Intel began releasing multicore chips for PCs and laptops two years ago. Last November, Intel released quad-core chips for PCs and servers.

Multicore chips improve performance by handling various parts of an application in parallel. Single-core chips, on the other hand, undertake tasks serially, said University of Southern California (USC) associate professor Mary Hall.

In some cases, programmers have used workarounds to enable PC games designed to run on single-core chips to run on multicore processors. For example, one type of workaround tries to assign each of multiple threads to a separate core on a chip, explained Martin Walker, a veteran multimedia software developer and chief technology offi-cer of game developer Artificial Mind & Movement.

However, Hall said, this doesn't work as well—and won't enable vendors to add new capabilities to their applications as easily—as designing programs from the beginning to run on multicore chips.

So far, though, vendors haven't moved fast enough to keep pace with the introduction of multicore chips, said Chad Marshall, an IT expert, author, and Bank of America's operational risk manager for technology and end-to-end processes.

One reason they haven't done so is that writing and rewriting programs is time-consuming and expensive, according to Dio De Niz, a member of the technical staff at the Software Engineering Institute, a federally funded R&D center based at Carnegie Mellon University. This is a particular issue for large software vendors with many legacy applications written to run on single-core chips.

Nonetheless, many vendors are already undertaking this task, including game developers, who are always interested in performance.

MULTICORE PROGRAMMING CHALLENGES

Sun Microsystems shipped the first multicore processor in 2000, said Marc Tremblay, the company's chief technical officer for micro-electronics. Sun designed the two-core MAJC (Microprocessor Architecture for Java Computing) chip to run 3D-graphics processing, video-conferencing, and general-purpose Java computing on PCs, but it never became commercially viable.

Before that, a number of high-performance computers and servers used multiple processors, which required applications designed to run in parallel. These programs have been on the market since the early 1990s, Tremblay noted.

Programming for single-core and multicore chips

For single-core chips, programmers write applications so that they prioritize tasks in the order they must be performed to do the designated activity most efficiently, explained De Niz.

The operating system then assigns the tasks to be run serially on the processor, said LSU's Sterling.

Developers write applications for multicore chips so that they will divide into tasks that the operating system can assign to run in parallel on multiple cores, he noted. And each core has its own multithreading capabilities and thus can divide its own tasks into parts that can run in parallel.

A key issue for programmers is how to divide up an activity into tasks. One common approach separates an activity into its various discrete subfunctions, said De Niz.

Developers also must figure out how to rewrite legacy programs built for single-core chips, a process that can be costly and time-consuming. In this case, developers must first determine which routines and tasks take up most of the compute
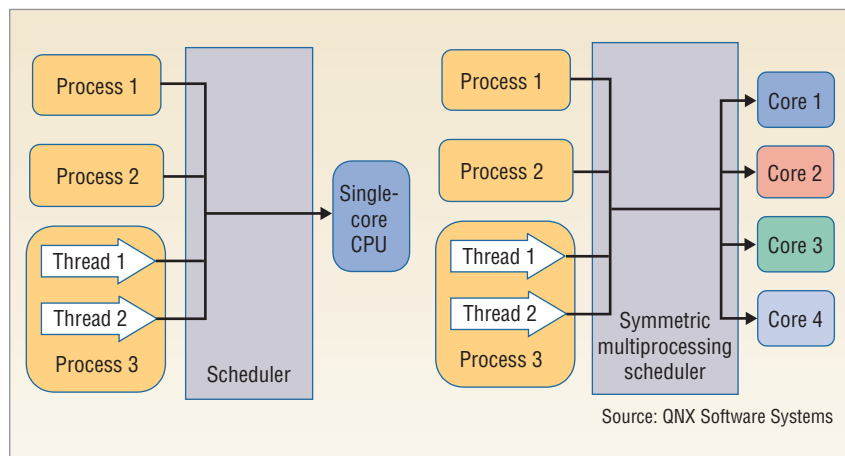
Figure 1. QNX Software Systems has a real-time operating system that offers the scheduling of tasks for single-core and multicore-chip applications. For single-core processors, the scheduler prioritizes tasks to be run serially on the chip. The symmetric multiprocessing scheduler prioritizes tasks too, but it also must determine how best to run them in parallel on a multicore chip's processing units.

time, explained Sterling. Then, he noted, they rewrite those tasks that can be easily parallelized. However, they often must leave other tasks as they were originally written.

Multicore challenges

While the goal of programming for multicore chips is clear, doing so isn't necessarily easy.

Dividing activities into smaller parts. A key concern for programmers is that some activities—such as those related to graphics—don't divide easily into subfunctions with naturally occurring beginning and ending points, De Niz said.

In these cases, the programmer must perform complex transformations that yield points at which an application can be broken into tasks that separate software into parts that are smaller and more granular than hard-to-divide subfunctions, or that change entire data structures.

Data dependency. Programmers must synchronize the chip's handling of an activity's various parts so that each calculation will have the access to the processed data from other parts that it needs to perform its work, an issue known as *data dependency*.

If synchronization requires more processing than is saved by paral-

lelizing the data, parallelism may not be useful.

Developers sometimes write *safety properties* into programs to ensure that applications don't try to run operations until they have the fully processed data they need from other operations and that they pass on only properly handled information, according to De Niz.

Data splitting. Data splitting occurs when, in the course of dividing a program for handling by multiple cores, two or more parts of the same data set are divided and sent to separate cores, according to De Niz. If, in the middle of a task, a computation needs part of a data set that is on another core, the results may turn out to be incorrect.

Balance. One of multicore developers' main goals is to maximize efficiency by splitting up tasks among different cores so that each is doing work of equal value to the overall process.

The programmers' effectiveness depends in part on their assumptions as to the relative value of subfunctions that run on separate cores. If one task designated to run as a separate subfunction turns out to be not all that important to the overall application, it might waste the resources of the core on which it is run.

This issue will become a bigger problem and thus create scalability issues as processors get more cores, according to Ray DePaul, CEO of RapidMind, a vendor of development tools for applications that run on multicore chips.

Testing. Applications that have multiple pieces running on different cores can have many execution paths. This makes testing a sufficient number of the paths difficult.

Programs that run serially, on the other hand, have fewer potential execution paths and thus are easier to test.

ANSWERING THE CALL

Various vendors are addressing the need to develop programs that run optimally on multicore chips.

For example, Microsoft designed Windows Vista to work efficiently with chips that have up to four cores. Apple's upcoming MacOS X v10.5 will also accommodate multicore processors.

QNX Software Systems—a provider of real-time operating system software, development tools, and services for embedded designs—has an RTOS that can effectively schedule the parts of an application that will run on multicore processors, as Figure 1 shows.

Intel has released tools—Intel Compiler 10.0 Professional Edition; Intel Thread Checker; Intel Thread Profiler; and Intel Thread Building Blocks for Windows, Linux, and the MacOS—to help developers write programs for the company's multicore chips.

AMD works with other companies and organizations—including Microsoft, the Portland Group, Sun, and the GNU open source community—to ensure their tools will be able to develop applications that run on AMD multicore chips.

Moreover, the vendor has designed its multicore chips to work with multiple brands of compilers, noted Margaret Lewis, AMD's director of commercial solutions.

The chipmaker has also provided

libraries—including the AMD Core Mathematical Library and AMD Performance Library—with highly optimized routines for developing, debugging, and optimizing applications that run on its multicore processors, she added.

RapidMind Development Platform parallelizes code that developers are working on, said DePaul. The platform comes with a runtime environment, APIs, and software libraries.

The developer can continue to implement in a familiar environment that looks single threaded. The code is then passed on via the APIs to the platform, which maps the work to all of the available cores, explained DePaul.

Some schools are preparing future developers to write applications for multicore processors. For example, Purdue University offers classes on this topic, noted associate professor Suresh Jagannathan.

NEW LANGUAGES FOR MULTICORE APPLICATIONS

Approaches that make multicore programming as simple as single-core, single-threaded programming will enable vendors to quickly get products to market, according to DePaul.

To accomplish this, some proponents say new programming languages may be necessary.

The new languages would let developers continue using the serial-programming approaches they are familiar with. However, the languages would automatically handle some of the manual choices developers currently must make to accommodate multicore programming, according to Jim Sexton, a staff member at IBM's T.J. Watson Research Center.

The Institute for Defense Analysis, a nonprofit R&D center that works for US government organizations, is developing such a language, Universal Parallel C, for the National Security Agency.

Sexton said IBM is working on the X10 language for writing applications that will run on multicore chips

in PCs, workstations, laptops, and supercomputers.

Many industry observers say multicore processors' future is bright. According to DePaul, 16-core chips will be available by the end of this decade. Intel has already developed a research chip with 80 cores.

Vendors won't have to rewrite applications that normally handle work serially, such as e-mail, word processing, and most traditional PC-based programs, because they won't benefit from parallelization.

For the same reason, they also won't have to rewrite software that doesn't use many resources, such as calculators, calendars, and other simple interactive utilities, noted USC's Hall.

According to Artificial Mind & Movement's Walker, most game developers have avoided the need to rewrite applications so far by building patches that let existing games run on multicore processors.

However, he explained, simply assigning each of the multiple threads to a separate core on a chip won't necessarily maximize the use of each core's capabilities and will require vendors to regularly rewrite patches to account for the increasing number of cores that chips will have.

Vendors will have an incentive now to rewrite CAD and other programs that are computation-intensive or those that can be easily parallelized such as database applications, for multicore chips.

And numerous developers of games and other types of programs are already writing their applications from scratch to work with multicore chips, a trend expected to continue in the future. ∎

David Geer is a freelance technology writer based in Ashtabula, Ohio. Contact him at david@geercom.com.