Yen-Kuang Chen,
Chaitali Chakrabarti,
Shuvra Bhattacharyya, and
Bruno Bougard

Signal Processing on Platforms with Multiple Cores: Part 1–Overview and Methodologies

Multicore processors are now prevalent in all major domains of signal processing. Many laptop and desktop computers today are shipped with dual-core and even quad-core processors. The number of cores is even higher for the Sony PlayStation 3, which is equipped with an eight-core IBM CELL Broadband Engine processor, Nvidia GeForce 9800 GX2, which has 256 stream processors, and SUN UltraSPARC T1/T2 processor, which has eight cores. Technology predictions indicate that this trend will continue and that the number of cores per processor can easily double around every two or three years.

The reason why multicore architectures are the vendors' choice today can be traced back to trends in silicon-processing technology. For several decades, technology scaling provided cheaper, faster, and more energy efficient transistors. For instance, in embedded systems, this provided an easy mechanism for achieving more computing performance and lower power consumption simultaneously. However, the "power wall" was hit at the 90-nm node. Since then, it has not been possible to increase performance at comparable power consumption levels only by technology scaling.

In high-performance computing, the trend was to increase the performance through higher clock speed at the cost of power consumption. For example, from the mid-1980s to the late 1990s, the power consumption of Intel's microprocessors doubled every two to three years and reached 20 W per square centimeter. Packaging solutions turned out to be more expensive than the integrated circuits themselves. It became imperative to pursue a different direction to increase performance.

Interestingly, for a given processor architecture in a given technology, the power consumption decreases faster than the performance when the clock rate is reduced. Typically, 20% under-clocking (with lower supply voltage) yields 50% power reduction and "only" 13% performance loss. For the same power consumption, a dual-core solution clocked at 20% less would bring, in theory, 73% more performance than a single core. This trend has led to a new approach in exploiting technology scaling, where the area cost reduction obtained from scaling is used to increase the number of cores.

THE REASON WHY MULTICORE ARCHITECTURES ARE THE VENDORS' CHOICE TODAY CAN BE TRACED BACK TO TRENDS IN SILICON-PROCESSING TECHNOLOGY.

While the challenges of designing multicore systems in hardware are many, writing efficient parallel applications that utilize the computing capability of many processing cores may require even more effort. To deliver the best performance, existing serial algorithms need to be redesigned to take advantage of the multicore computing power. This is because the best sequential algorithm is not necessarily the best parallel algorithm.

Signal processing algorithm designers of the future will need to better understand the nuances of multicore computing engines. Only then can the tremendous computing power that such platforms provide be harnessed to their full potential.

To give a thorough view of the area, we offer two special issues on this topic. This first special issue is aimed at providing coverage of key trends and emerging directions in architectures, design methods, software tools, and application development for design and implementation of multicore signal-processing systems. A follow-up of this issue will describe novel applications that can be enabled by platforms with multiple cores, and more extensive design examples of signal processing on platforms with multiple cores that demonstrate useful techniques for developing efficient implementations.

There are a total of 11 articles in this issue. These span three thrust areas: architectures (articles 1–3), software tools and methodologies (articles 4–7), and design examples (articles 8–11). Together, these articles provide the breadth needed for a casual reader and the depth needed for a digital signal processing practitioner.

The first architecture article by Blake et al. is on general-purpose multicore architectures that can be used from laptops and desktops to servers. It describes the key attributes, which include power/performance, processing elements, memory systems, and application domains, that are common to all multicore processor implementations, and then illustrates these attributes with current and future multicore designs.

Karam et al. present an overview of existing multicore DSP architectures. At the hardware level, the architectures

Digital Object Identifier 10.1109/MSP.2009.934556

are classified based on memory hierarchy and interconnect. The article also describes existing software tools and emerging applications based on these architectures.

Wolf surveys multiprocessor system-on-chip (MPSoC) systems that were developed to meet the needs of embedded signal processing and multimedia. These architectures are mostly heterogeneous and pose special software challenges due to their combination of parallelism and heterogeneity. A historical perspective is provided. To utilize the computing power of multicore systems, DSP software tools and methodologies have to be reexamined to provide effective exploration of parallel-processing solutions, and address novel constraints associated with multicore software implementation.

Mehrara et al. make the case that software compilation tools that find and exploit different types of parallelism are necessary for the success of multicore systems. The article describes existing compilation tools and strategies for both static and dynamic compilation.

The next article, by Haid et al. analyzes the major challenges in MPSoC software development and show that typical software design flows fail to support design-space exploration or software synthesis in a way that is suitable for multicore signal-processing systems. The article then advocates the use of design flows based on formal models of computation and focuses on application of the Kahn process network model to demonstrate the benefits of such an approach.

Park et al. survey a broad range of design methods and tools for software development on MPSoC architectures. Four different approaches are analyzed: the compiler-based approach, language-extension approach, model-based approach, and platform-based approach.

Kim and Bond describe key software technologies including parallel-programming languages for developing applications on multicores. It also proposes a software development process for high-productivity development of signal-processing applications on multicore platforms.

The next set of articles describes successful mapping of key signal-processing algorithms onto multicore platforms.

[SIGNAL PROCESSING ALGORITHM DESIGNERS OF THE FUTURE WILL NEED TO BETTER UNDERSTAND THE NUANCES OF MULTICORE COMPUTING ENGINES.]

Franchetti et al. describe a framework that enables automation of discrete Fourier transform implementation for multicore platforms. They describe optimizations, all derived using Kroenecker-product formalisms, to address the challenges of parallelization, vectorization, and memory hierarchy.

Lin et al. describe techniques for parallelizing video-processing kernels for multicore platforms. The article provides an overview of basic parallel-programming concepts and technologies and describes optimization techniques through multiple examples.

The next article, by Amer et al., also on video processing, provides an overview of reconfigurable video coding (RVC), and how it can be mapped to multicore architectures. The authors demonstrate that while RVC automatically provides flexibility and reconfigurability, the formulation of RVC functionality in terms of dataflow graphs facilitates efficient mapping onto multicore platforms.

The article by You et al. closes the issue by presenting a scalable inference engine for large vocabulary continuous speech recognition. The authors explore four application-level implementation alternatives on two parallel platforms. They demonstrate that different algorithms may need to be explored to deliver the best performance on different platforms.

We would like to thank Shih-Fu Chang, Li Deng, Dan Schonfeld, and Doug Williams for their encouragement of this special issue project. We sincerely thank the authors for their valuable contributions, and to the anonymous reviewers for their help in ensuring the quality of this special issue. We would also like to thank everyone who submitted white papers to this special issue and express our regret that, due to limited space and the need for balanced coverage, not all high-quality proposals could be encouraged for further development.

We hope that you enjoy the articles in this special issue of *IEEE Signal Processing Magazine* and find its contents informative and useful for the overview of the trends and challenges of signal processing on systems with multiple/many cores. Please stay tuned for the follow-up March 2010 issue for novel applications on multiple cores and more extensive design examples. [SP]