The academic programming language community continues to reject the change in programming practices brought about by scripting, remaining deaf to John K. Ousterhout’s assertion that scripting can provide higher-level programming for the 21st Century.

That scripting has developed in the shadow of object-oriented programming explains part of the problem. The two can be compatible, but one philosophy has received the most attention, with scripting appearing language by language. Ousterhout declared scripting on the rise, but perhaps so too are programming language pragmatics.

In Praise of Scripting: Real Programming Pragmatism
pp. 22-26
Ronald P. Loui

Efficient Embedded Computing
pp. 27-32
William J. Dally, James Balfour, David Black-Shaffer, James Chen, R. Curtis Harting, Vishal Parikh, Jongsoo Park, and David Sheffield

Embedded computing applications demand both efficiency and flexibility: The bulk of computation today happens not in desktops, laptops, or data centers, but in embedded media devices. More than a billion cell phones are sold each year, and a 3G cell phone performs more operations per second than a typical desktop CPU.

Demanding performance and efficiency requirements drive most media devices to perform their computations with hardwired logic in the form of an application-specific integrated circuit. While ASICs meet embedded applications’ energy-efficiency demands, they are difficult to design and inflexible. Increasingly, embedded applications demand both flexibility and efficiency.

In warp processing, a compute platform transparently performs FPGA circuit compilation as a program’s binary executes on a microprocessor.

Automating Postsilicon Debugging and Repair
pp. 47-54
Kai-bui Chang, Igor L. Markov, and Valeria Bertacco

Increasing semiconductor design complexity lets more errors escape presilicon verification, to be discovered only later in prototype chips. While most steps in the IC design flow are highly automated, researchers have devoted little effort to the postsilicon debugging process, making it difficult and ad hoc.

The author’s proposed FogClear methodology, powered by novel techniques that integrate logical, spatial, and electrical considerations, systematically automates this process. Empirical results indicate that FogClear’s key components—PAFER, PARSyn, SymWire, and SafeRe-synth—repair numerous functional and electrical errors in most benchmarks, demonstrating their effectiveness in postsilicon debugging.

SIISAM: A Model for Secure Inhomogeneous Information Systems
pp. 56-61
Kun Wang, Zhonghai Yin, Libua Zhou, Feng Yuan, and Zengxin Li

Researchers in China’s E-Government Experimental and Demonstration Project have created the secure inhomogeneous information system architecture model. SIISAM provides security-critical services as well as local fault and disaster recovery to ensure continuous service; uses Web service technology to support interoperability and extensibility; and is a layered rather than a process-driven model that can more flexibly accommodate changes in applications, thereby facilitating system design, development, and deployment.

Amdahl’s Law in the Multicore Era
pp. 33-38
Mark D. Hill and Michael R. Marty

Computations might execute faster as circuits on a field-programmable gate array (FPGA) than as sequential instructions on a microprocessor because a circuit allows concurrency, from the bit to the process level. Several tools seek to compile popular microprocessor-oriented software programming languages to FPGAs. Key barriers to adoption include the difficulty of integrating such tools into established microprocessor software development flows and such tools’ nonconformance to the standard binary concept that forms the basis of many computing domains’ ecosystems.

In warp processing, a compute platform transparently performs FPGA circuit compilation as a program’s binary executes on a microprocessor.