



**> XMOS** David May

## A MULTICORE, PARALLEL approach to computing

*XMOS puts multiple processing cores, each with its own memory and I/O system, on a single chip.*

DAVID MAY is chief technical officer of fabless-semiconductor start-up XMOS, based in Bristol, England. Long an advocate of multicore, parallel computing, he was an early recruit to INMOS in 1978, where he was the architect of the Transputer and the author of the Occam language. STMicroelectronics eventually absorbed INMOS and many elements of the Transputer into the ST20 architecture. May later became head of computer science at the University of Bristol, then a co-founder of XMOS.

May originally studied at Cambridge University. "I was one of the first computer-science graduates," he says. His involvement with computation long predated university studies, however. "I was always building and inventing things and first built a calculator based on relays at age 11," says May. By the early 1970s, he was working with robotic systems, and the first microprocessors emerged. "It was immediately apparent how they could work coupled together; it became the first application of distributed computing," he recalls.

Although, in those days, accurately controlling a motor with feedback used the power of a complete computer, May recalls the "obvious" concept of localized control programs with closed feedback loops passing messages around a wider system and mimicking biological systems. This sort of broad-based innovation process appeals to May. In contrast, he says, alluding broadly to the gigahertz race for ever-faster single-processor cores, "Much of what went on in the 1990s wasn't innovation."

Around 2000, May says, it became apparent that there was an opportunity to undertake a new venture, once more based on the ideas of multiple processing cores and communicating simultaneous processes. The XMOS architecture places a number of general-purpose processing cores, each with its own memory and I/O system, on a single chip. The cores have direct support for concurrent processing (multithreading), communication, and I/O. Any processing thread can communicate with any other thread over a fast intrachip-switch fabric or interchip serial links. The architecture supports any language; XMOS added extensions to form XC, a version of C that supports I/O, multicore, and precision timing.

May emphasizes that XMOS' architecture is not a reborn Transputer—although it has more in common with the Transputer than with any other earlier proces-

sor. He views the innovation process as one that examines real-world systems and creates analogies of them with processors or with technology. It is also one that is unafraid of reaching back into the past to use or reuse ideas. Despite his academic tenure, however, May

says, "I want to see something result from it. Work that ends in an academic paper alone—that's not me."

Perhaps with that thought in mind, May looks back wryly to the innovations associated with the INMOS venture: "Everything was at or beyond the limit. We were pushing the [process] technology as far as we could to get as much on a single die as possible; [Occam] was completely new; the programming model was new. If anything, there was an excess of innovation." By contrast, he says the XMOS venture recognizes that, if it is to capture the attention of the wider base of embedded-computing designers, programming for the XMOS chip must be as familiar as possible. "It looks like C," he says.

May also notes the importance of timing. One of the reasons he feels that the time is right for a switch to a parallel model of embedded computing is that innovations are appearing in a range of disciplines, not only the computing domain. Referring once again to robotics, he notes that the emergence of strong, lightweight materials changes the picture. You can not only precisely compute all of the necessary control actions to perform robotic actions but also execute them in a mechanical environment with low inertia and feasible power levels. He continues to be an advocate of creating systems that are appropriate to the task at hand and no more: "Think lean and mean; small systems and low power are very important."

May aspires to foster the innovation process by making it simple, once again, to quickly and easily explore new concepts by directly prototyping and building things. He would like to see a processor-based environment become "the TTL of the next two or three decades," with the capability for "mass personalization and the potential to produce mass differentiation" leading to a cascade of new and useful products.

—by Graham Prophet

MAY VIEWS  
THE INNOVATION  
PROCESS AS ONE  
THAT EXAMINES  
REAL-WORLD  
SYSTEMS AND  
CREATES  
ANALOGIES  
OF THEM WITH  
PROCESSORS OR  
WITH TECHNOLOGY.