DOI:10.1145/2735839

To view the accompanying paper, visit doi.acm.org/10.1145/2735841 rh

Technical Perspective
The Specialization Trend in Computer Hardware

By Trevor Mudge

THE BREAKDOWN OF Dennard scaling has become a new challenge that computer architects face as we move to smaller technology nodes. In the past, Dennard scaling meant that speed increased, power dropped, and area shrank. And, following Moore's Law, this triple win was achieved without appreciable increase in chip cost. This remarkable era in which Moore's Law held sway is drawing to a close for CMOS-based technology, and, as we transition to sub-20nm feature sizes, at least two of these improvements—speed and power—are grinding to a halt. In fact, several pundits have argued the 28nm node may be the cheapest node in terms of cost per transistor.

This slowdown in scaling will have a profound impact across the whole spectrum of computing, starting with the manner in which hardware systems are architected in a future where increased functionality and much tighter power constraints will be the norm. Nowhere is this more apparent than in mobile platforms, particularly smartphones, where there are ever-increasing demands for functionality without compromising battery life. It will no longer always be possible to simply move to the next-generation general-purpose computer to meet tighter energy-performance constraints.

An alternative that improves energy-efficiency is specialization, but it only makes economic sense if there is significant demand. A balance can often be found by designing application-domain-specific components that have a degree of programmability, which is enough to open up significant application space. The work presented in the following paper does just that.

The authors present a programmable convolution engine that takes a 1D or 2D template (or stencil) and convolves it with a 1D or 2D array of data—a 2D image is a typical example. The stencil is much smaller than the image. It can be as small as 3x3 pixels, whereas the image may have hundreds of millions of pixels. Its size is programmable, as is the function it performs. The applicability of this kind of operation is widespread, particularly in image processing, computer vision, and the emerging field of virtual reality. The rapid adoption of such functions into mobile platforms underscores the need for them to be performed in a power-efficient way: it is not difficult to justify a specialized processor for convolution. This notion was the impetus behind the authors' investigation into a programmable convolution engine.

The authors present a detailed study to show where their design fits in the space between fixed-function hardware and general-purpose hardware.

Within the space of programmable convolution engines there are still important design choices to be made in order to make the most of the decisions to specialize. To guide these choices the authors make the crucial observation that "specialized units achieve most of their efficiency gains by tuning data storage structures to the data-flow and data-locality requirements of the algorithm." This important observation greatly benefits power-performance efficiency. Convolution involves a great deal of data movement, for example, a large image must be accessed from memory. These accesses follow well-defined patterns that can be precomputed once the particular type of convolution and the size of the stencil are known. Thoughtful design allows these access patterns to be performed so as to minimize redundant accesses, greatly reducing power consumption while speeding the calculation.

To conclude, the authors present a detailed study to show where their design fits in the space between fixed-function hardware and general-purpose hardware. Specifically, they consider two ends of the spectrum: a custom implementation and a general-purpose processor augmented with SIMD instructions. The SIMD instructions model similar features found in commercial processors, such as Intel's Streaming SIMD Extensions and ARM's NEON extensions. They show their convolution engine implementation is 8–15 times more energy and area efficient than a general-purpose processor plus SIMD, but within a factor of 2–3 times of the custom unit. This analysis neatly illustrates the pros and cons of specialization with some degree of programmability. Finally, the analysis alone is worth study because it provides one of the clearest examinations of where energy and area are expended in today's processors. C

Trevor Mudge (tnm@eecs.umich.edu) is the Bredt Family Professor of Engineering in the EECS Department at the University of Michigan, Ann Arbor.

Copyright held by author.