The Future in Your Pocket

Patrick Crowley
Applied Research Laboratory
Department of Computer Science & Engineering
Washington University
St. Louis, MO USA
pcrowley@wustl.edu

This article is an editorial note submitted to CCR. It has not been peer reviewed.
Authors take full responsibility for this article's technical content. Comments can be posted through CCR Online.

ABSTRACT
There is a growing sentiment among academics in computing that a shift to multicore processors in commodity computers will demand that all programmers become parallel programmers. This is because future general-purpose processors are not likely to improve the performance of a single thread of execution; instead, the presence of multiple processor cores on a CPU will improve the performance of groups of threads. In this article, I argue that there is another trend underway, namely *integration*, which will have a greater near-term impact on developers of system software and applications. This integration, and its likely impact on general-purpose computers, is clearly illustrated in the architecture of modern mobile phones.

Categories and Subject Descriptors
C.1.4 [Processor Architectures]: Parallel Architectures – *Mobile processors*, C.2.m [Computer-Communication Networks]: Miscellaneous

General Terms
Performance, Design, Economics

Keywords
Mobile phones, multicore, CPU

1. INTRODUCTION
Suppose one wanted to be prepared to make good use of the next generation of computers. What should one do? There has been much talk over the past few years of the rise of multicore processors and of the need for parallel, multi-threaded programming in order to make use of them. So, to be prepared for the future, one might choose to brush up on the latest in parallel programming. In this article, however, I offer different advice. I say: to be prepared for the future, start programming your mobile phone.

2. MULTICORE TO THE MASSES?
Are we all destined to become parallel programmers? We have been warned that CPU clock frequencies (i.e., the rate of operation of basic machine instructions) will no longer increase significantly between processor generations and will thus no longer provide consistent performance gains for the software we use today.

The technical motivations for multicore processors are sound. Adding multiple cores to a CPU increases the peak instruction bandwidth without requiring an increase in clock frequency. This is important because dramatic increases in operating frequency must be avoided in order to keep thermal design power reasonable for general-purpose computing platforms. Indeed, merely maintaining current processor frequencies has required substantial semiconductor innovation [1]. As a simple example, two cores operating at 2 GHz have a combined instruction completion bandwidth equivalent to a single 4 GHz processor core. However, two concurrent programs, or threads of execution, are needed to realize the increased performance. If the program you care about is a single thread, then you will see only indirect benefits when moving to a CPU with more cores. These indirect benefits are due to reduced sharing; your program will share its core with fewer other programs, and hence might experience some speedup.

Today's general-purpose processors feature 2 or 4 processor cores. Will tomorrow's CPUs feature tens or hundreds of cores? The future is unclear, and will very much depend on how well various application domains can leverage multiple processor cores.

Some domains area already dominated by multicore-friendly, thread-parallel software architectures. If your favorite software runs on a cluster or compute cloud, then you are already in a strong position to benefit from multicore processors (and, happily, you do not really care about the details of a single computer). Perhaps most extreme are those illegal, decentralized systems such as botnets which manage to make efficient use of large-scale parallel resources without the benefit of legal access to the individual computers!

At a lower level of system abstraction, high-performance embedded processors such as network processors have leveraged multiple cores to meet I/O intensive real-time constraints. Cisco, for example, has designed a 192-core processor for its high-end router line-cards [2]; each line-card features two of these processors and they have been shipping since 2004.

But what of other application domains? What about the software that runs on PCs and is developed by the largest segment of the software engineering community? It is less clear that general, PC-

based applications can be structured to exploit increasing numbers of processor cores.

There are good technical reasons to be skeptical of the arrival of commodity CPUs with large numbers of cores. One might ask, where is greater throughput needed? The rising use of platform virtualization argues that many existing server platforms are under-utilized rather than resource constrained. How will pin bandwidth per core change over time? The performance of most applications is memory-bound, so reducing off-chip bandwidth per-core may not be a uniform advantage across applications.

There are also sound non-technical reasons to be skeptical and to consider alternatives. First, assume for a moment that you are a multicore processor architect and that you are explaining yourself to your Grandmother (grandparents may not all be technically savvy, but they are a significant and growing percentage of the consumer population in North America, Western Europe, and elsewhere).

> **Dialogue with Grandmother**
>
> G: "Remind me, what do you do for a living?"
>
> *"I study multicore processors. They will allow your next PC to have 2, 4, 8, or 16 computers inside."*
>
> G: "I only need one. Can I buy one for 1/2, 1/4, 1/8, or 1/16 the price?"
>
> *"Well…"*

This becomes a troublesome conversation, so my inclination is to move on to someone who would not require so much explanation of technical details. Let us now imagine a conversation with a purchaser of IT infrastructure.

> **Dialogue with Director of IT**
>
> *"The next generation of server chips will be 16-core processors. They will allow your server box/blade to have 16 computers inside."*
>
> IT buyer: "16 cores? Great, I was going to buy 160 server machines next year. Instead, I'll buy 10—or perhaps 20 just in case."

This feels a bit like a generic argument about buying faster computers. However, the availability of multiple cores complicates the purchasing decision, at least as compared to the bygone days of increasing clock frequency. It prompts the buyer to think: "how many cores do I need?" Which in turn begs the question of how well utilized the current number of cores are. These are new sorts of questions for buyers of computers.

It is reasonable, and useful, to ask if there is an alternative. To motivate one particular alternative to the aggressive multicore future envisioned by some, let us consider a fictional historical analogy.

Consider an ALU chip vendor at the dawn of the VLSI era. Prior to the age of VLSI, computer vendors assembled digital computer systems with discrete components—register file chips, ALU chips, controllers, etc.—which consisted of at most hundreds or a few thousand transistors. With VLSI, however, greatly increased numbers of transistors became available for use on single chips. What should the ALU vendor do with 10s and 100s of thousands of transistors? Two options seem clear.

> **Option 1**: Multi-ALU to the masses! To make the design challenge feasible, ALU chips could be scaled to double the number of ALU cores per chip every two years or so. Making good use of these cores may be a challenge to computer designers, but there may be no alternative. Unless we consider…
>
> **Option 2**: Greater integration: move register files, data paths, and controllers on-chip. Rather than designing ALUs, the former ALU vendor can begin offering processor chips, i.e., CPUs.

Of course, this is a fictional analogy because it is unlikely that any firm seriously considered aggressively scaling the numbers of ALUs per chip (and the definition of a digital computer wasn't so well understood prior to VLSI). However, viewed in this way, aggressive *platform integration* can be seen as a viable alternative to an aggressive multicore CPU roadmap. Not sure what to do with your next doubling of transistors? Integrate the graphics processing unit (GPU) or other performance-critical hardware accelerators. After that, integrate a large chunk of main memory. If this direction is taken, perhaps today's CPU vendors will over the next few years become "computer chip" or system-on-a-chip vendors. Naturally, there would be strong impacts on the current relationships between semiconductor and computer vendors, but that is not necessarily a bad thing.

3. WHY INTEGRATE MORE COMPONENTS?

Other than making use of abundant transistors, what benefit is there in integrating system functionality onto one chip? There is an obvious benefit for the size of computing devices, since a smaller number of chips fit within smaller form factors. Also, for tasks that use an integrated component, one can expect improved:

- performance,
- power efficiency, and
- area efficiency.

Each of these are due to shortened, more efficient interconnect paths, and the resulting shorter latencies and greater bandwidths. It takes far less time, die area, and power to drive an on-chip channel than an off-chip one.

Additionally, users might find unexpected uses for integrated components. At least within the academic community, there is no shortage of people using GPUs for solving non-graphics problems.

For current general-purpose CPU vendors, the term "multicore" has merit for marketing purposes, but I feel that it obscures the primary issue. The critical issue is to decide what to integrate: general-purpose cores, special-purpose cores, or some combination?

There is good evidence that CPU vendors are already thinking in this way, despite their apparent preference for characterizing a multicore future. AMD, which already provides CPUs with integrated memory controllers and high-performance I/O paths,

has announced plans to integrate a GPU on-die with a CPU [3]. Intel while historically eschewing integration—only recently has Intel announced IA-based products with integrated memory controllers—has announced two product lines which are unique for their degree of platform integration. Intel's Atom product line, previously known as Silverthorne and its chipset Poulsbo, is a low-power, highly-integrated processor meant for embedded consumer electronic devices such as in-car entertainment systems (as well as mobile Internet devices, which Intel has described but do not yet exist as a distinct product market). Tolapai [5] is another low-power chip with integrated I/O and cryptography accelerators.

For now, the CPU vendors have plans for integrated chips, but the chips are not yet available. Alternatively, mobile phones are a computing platform that years ago compelled technology suppliers to confront the integration question directly. If you have not had a reason to learn about the organization of processing resources in mobile phones, you may be in for a surprise.

4. WHAT CAN BE LEARNED WITH MOBILE PHONES?

Modern mobile phones, particularly high-end devices sometimes termed smart phones, are built around highly-integrated multicore systems-on-a-chip. Figure 1 illustrates the types of units found in a typical chip, such as the OMAP processor family from TI [6].

In addition to a primary CPU (often an ARM-based, superscalar processor), these chips integrate a number of special-purpose cores: 2D/3D graphics accelerators; DSPs for images, video, and audio; cryptography units for bulk encryption and authentication; digital display controllers for small integrated displays as well as external TVs and monitors; controllers for a variety of radio types, including mobile phone networks, WiFi, Bluetooth, GPS, and digital television; controllers for still and video cameras; controllers for microphones and speakers; codecs and controllers for non-traditional user I/O such as speech recognition and synthesis and touch panels; controllers for a variety of memory types, including DRAM, NOR/NAND flash, and external non-volatile memory cards; as well as controllers for traditional I/O channels such as USB and Firewire. All of these components can be found in single-chip solutions that cost less than $10 per unit in large volumes.

In addition to a rich integration, many platform characteristics of current mobile phones are highly relevant to future general purpose computing devices. These characteristics include:

- Mobility and location awareness in physical space and between networks.
- Ability to connect to several distinct network types.
- Integrated platform support for capturing and replaying different media types, including audio and video.
- A large and growing installed base of systems.

The application development environments and platform APIs for mobile phones are designed to enable access to integrated features and ease the use of these functional characteristics in software and services. As a result, they are quite a bit different, and perhaps more forward-thinking, than the APIs provided by general-purpose operating systems.

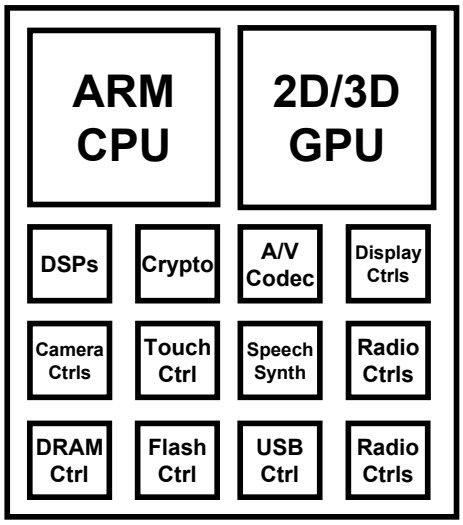

Figure 1. Organization of a typical mobile phone processor.

An additional qualitative property is that mobile phone environments often require software developers to deal explicitly with varying platform characteristics, such as battery life, available network connections, available displays, available hardware accelerators, and so forth. Providing sustainable application-level support for these differing characteristics is likely to be an important part of future general-purpose programming environments, as software is developed with a greater emphasis on power efficiency and platform independence.

5. WILL THERE BE ONE DOMINANT PLATFORM?

For all their virtues, mobile phones feature user interfaces that are dramatically constrained for many tasks as compared to PCs and laptops. Whenever a keyboard and large display are useful, a mobile phone is not likely to be an ideal platform. So no one is suggesting that the mobile phone form factor will displace the laptop or desktop form factor. But there are good reasons to consider the possibility that mobile phone technology components, such as processors, may displace PC components.

Historically, PCs due to their wide volumes and general-purpose nature have been the foundation for IT economics. Recently, however, mobile phone platforms have shown PC-like capabilities, and four-fold greater sales volumes. In 2006, for example, 230M computers were sold as compared to 960M mobile phones (based on estimates found in the 2006 annual reports from Intel, AMD, Nokia, and Motorola).

It is also natural to ask which platforms host more application innovation today, and what the trend is over time. Certainly mobile phones have experienced an explosion of platform features over the past five or so years. For many users, the mobile phone serves as a primary web browser, email client, instant messenger, and digital camera. PCs are unquestionably the primary *development* platform, but it is not clear that they will be the primary *application* platform in the future.

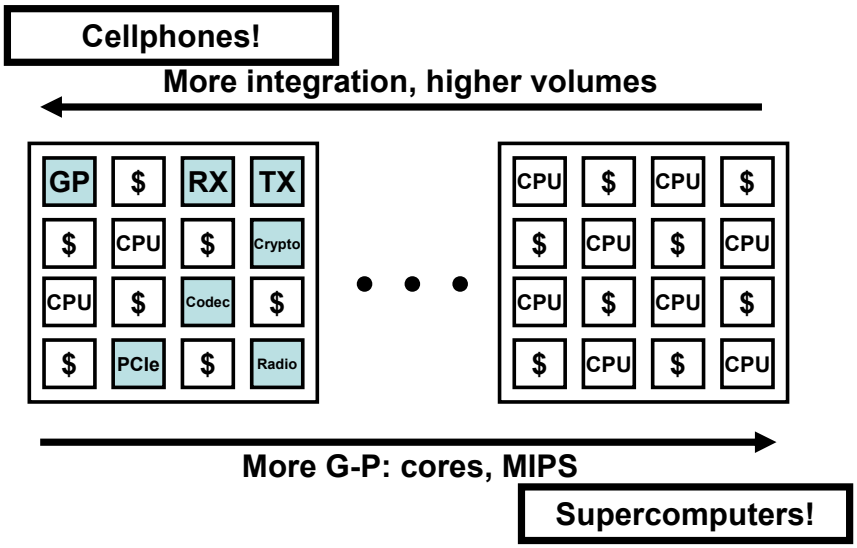

Figure 2. A family of chips along a spectrum of platform integration. In the figure, '$' stands for on-chip cache memory.

6. CONCLUSION

In my view, if you want to be prepared to program the computer of the future, you can start today with your mobile phone. Based on my own experience, owners of Nokia Series 60 phones can very easily explore the landscape with the Python for Nokia [7] open-source software package.

While this discussion has mostly focused on the positive aspects, platform integration poses challenges, especially for open systems software. Proprietary hardware blocks in embedded systems typically ship with proprietary software and drivers, which are licensed by the software or platform integrator. This poses a challenge for software developers who hope to exploit the low-level capabilities of such components, or who hope to develop their own operating systems or other pieces of system software. As general-purpose systems embrace greater levels of platform integration, this issue may threaten the open nature of general-purpose platforms.

Figure 2 illustrates the family of chips one might expect to see from the major CPU vendors in the near future. Such a family represents a spectrum of chips, suited for different classes of computing devices, with those suitable for platforms like mobile phones on the left and fully-general high-performance supercomputer chips on the right. If you believe that the highest-volume computing platforms are the most relevant ones, then you might also feel that 'supercomputer' is a euphemism for a system that only exists via government subsidy. Opinions differ as to which end of the spectrum is most significant.

Of course, mobile phones are not the only high-volume computing platform with relevance for the future. As programmable digital TVs [8], programmable set-top boxes, and mobile Internet devices arrive in volume, they may emerge as dominant application platforms. In my view, programming these systems will be more like programming highly-integrated mobile phones than general-purpose multicore processors.

Finally, this article has presented my personal view. In the interest of full disclosure, I must point out that it is a substantially contrarian one! Fortunately, however right or wrong my opinion may be, Moore's Law promises that it will not be long before we see how the significance of platform integration compares to that of multiple cores in next-generation computers.

7. ACKNOWLEDGMENTS

I would like to thank Srinivasan Keshav for encouraging me to put these thoughts down on paper. I first discussed this topic publicly at a workshop organized in late 2006 by the Intel Research Council, and I would like to thank Erik Johnson for inviting me to speak at the workshop and for giving me a reason to organize this perspective.

8. REFERENCES

[1] Bohr, M.T., Chau, R.S., Ghani, T., Mistry, K., "The High-k Solution," Spectrum, IEEE, 44, 10 (Oct. 2007), pp.29-35.

[2] Cisco Systems. Silicon Packet Processor in CRS-1 Router. http://www.cisco.com/en/US/products/ps5763/index.html

[3] AMD. AMD Completes ATI Acquisition and Creates Processing Powerhouse. AMD press release, Oct 25, 2006.

[4] Intel. Intel Announces Intel® Atom™ Brand for New Family of Low-Power Processors. Intel press release, Mar 2, 2008.

[5] Intel. Intel Introduces Future VPN Solution: Tolapai. http://www.intel.com/design/intarch/demos/soc/demo.htm

[6] Texas Instruments. TI OMAP Technology. http://focus.ti.com/omap/docs/omaphomepage.tsp

[7] Nokia. Python for S60. http://opensource.nokia.com/projects/pythonfors60/

[8] Cable Television Laboratory. Tru2way, formerly OCAP. http://www.tru2way.com