Embedded Parallel Computing
2011
Seminar Topics
Our research addresses a key technological problem for microprocessor architects: How to leverage growing quantities of chip resources even as wire delays become substantial.

In an interesting twist, just as the clock frequency of processors has risen exponentially over the years, the fraction of the chip that is reachable by a signal in a single clock cycle has decreased exponentially.¹
The Raw processor design divides the usable silicon area into 16 identical, programmable tiles. Each tile contains

- one static communication router;
- two dynamic communication routers;
- an eight-stage, in-order, single-issue, MIPS-style processor;
- a four-stage, pipelined, floating-point unit;
- a 32-Kbyte data cache; and
- 96 Kbytes of software-managed instruction cache.

Future Raw processors will have hundreds or perhaps thousands of tiles.
Application mapping
The operating system allocates a rectangular-shaped number of tiles proportional to the amount of computation that is required by that process.
Topic 2 – Energy Efficient Computing for Mobile Applications

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

A cell phone’s computing challenges are even more impressive when we consider that these performance levels must be achieved in a small handheld package with a maximum power dissipation of about 1W.

Mobile Supercomputers for the Next-Generation Cell Phone

Mark Woh, Scott Mahlke, and Trevor Mudge, University of Michigan
Chaitali Chakrabarti, Arizona State University

Published by the IEEE Computer Society
The International Telecommunications Union has proposed fourth-generation (4G) wireless technology to increase bandwidth to maximum data rates of 100 Mbps for high-mobility situations and 1 Gbps for stationary and low-mobility scenarios like Internet hot spots (www.ieee802.org/secmail/pdf00204.pdf). This translates into an increase in computational requirements of 10 to 1,000 times over previous third-generation (3G) wireless technologies, with a power budget of approximately 1 W for all the computation. Other forms of signal processing, such as high-definition video, are also up to 100 times more compute-intensive than current mobile video.

*AnySP* seeks to create a fully programmable architecture that supports 4G wireless communication and HD video decoding.
Figure 1. Peak processing throughputs and power budgets of 3G and 4G protocols. Conventional processors cannot meet these power-throughput requirements.
AnySP seeks to create a fully programmable architecture that supports 4G wireless communication and HD video decoding.

Figure 2. AnySP processing element. This processing element architecture consists of integrated SIMD and scalar data paths.
Nvidia’s Compute Unified Device Architecture (CUDA) is a software platform for massively parallel high-performance computing on the company’s powerful GPUs.
Figure 2. Nvidia GeForce 8 graphics-processor architecture. This particular GeForce 8 GPU has 128 thread processors, also known as stream processors. (Graphics programmers call them “programmable pixel shaders.”) Each thread processor has a single-precision FPU and 1,024 registers, 32 bits wide. Each cluster of eight thread processors has 16KB of shared local memory supporting parallel data accesses. A hardware thread-execution manager automatically issues threads to the processors without requiring programmers to write explicitly threaded code. The maximum number of concurrent threads is 12,288. Nvidia GPUs with 128 thread processors include the GeForce 8800 GTX, GeForce 8800 Ultra, and GeForce 8800 GTS 512MB; the Quadra FX 5600; and the Tesla C870, D870, and S870.
Parallel threads/streams

A few years ago, pioneering programmers discovered that GPUs could be reharnessed for tasks other than graphics. However, their improvised programming model was clumsy, and the programmable pixel shaders on the chips weren’t the ideal engines for general purpose computing. Nvidia has seized upon this opportunity to create a better programming model and to improve the shaders. In fact, for the high-performance computing market, Nvidia now prefers to call the shaders “stream processors” or “thread processors.”

Occasionally, of course, a thread does need to access off-chip memory, such as when loading the off-chip data it needs into local memory. In the CUDA model, off-chip memory accesses usually don’t stall a thread processor. Instead, the stalled thread enters an inactive queue and is replaced by another thread that’s ready to execute. When the stalled thread’s data becomes available, the thread enters another queue that signals it’s ready to go. Groups of threads take turn executing in round-robin fashion, ensuring that each thread gets execution time without delaying other threads.

The processors can switch their attention among these threads in a single clock cycle. All this run-time thread management is transparent to the programmer.

Microprocessor Report 2008
From Microprocessors to Nanostores: Rethinking Data-Centric Systems

Parthasarathy Ranganathan, HP Labs

The confluence of emerging technologies and new data-centric workloads offers a unique opportunity to rethink traditional system architectures and memory hierarchies in future designs.

significant advances in scalability as well as innovations in the software stack. Looking further out, emerging technologies such as photonics, nonvolatile memory, 3D stacking, and new data-centric workloads offer compelling new opportunities. The confluence of these trends motivates a rethinking of the basic systems' building blocks of the future and a likely new design approach called nanostores that focus on...
Interestingly, datasets and the need to operate on larger fractions of the data in-memory continue to increase, there will likely be an inflection point at which conventional system architectures based on faster and more powerful processors and ever deeper memory hierarchies are not likely to work from an energy perspective (Figure A). Indeed, a recent exascale report identifies the amount of energy consumed in transporting data across different levels as a key limiting factor. Complex power-hungry processors also are sometimes a mismatch with data-intensive workloads, leading to further energy inefficiencies.
NANOSTORES: A NEW SYSTEM ARCHITECTURE BUILDING BLOCK?

The confluence of these various trends—future large-scale distributed data-centric workloads with I/O-intensive behavior, innovations in the software stack, and the emergence of new nonvolatile memories potentially timed with the end of scaling for DRAM—offers a unique opportunity to rethink traditional system architectures and memory hierarchies in future designs.

Nanostores offer one such intuitive, and potentially advantageous, way to leverage this confluence of application and technology trends. We coined the term nanostores as a duality of microprocessors to reflect the evolution to nanotechnology and the emphasis on data instead of compute. The key property of nanostores is the colocation of processors with nonvolatile storage, eliminating many intervening levels of the storage hierarchy. All data is stored in a single-level nonvolatile memory datastore that replaces traditional disk and DRAM layers—disk use is relegated to archival backups.
Figure 3. Nanostores colocate processors and nonvolatile memory on the same chip and connect to one another to form a larger cluster for data-centric workloads.
Implantable medical devices have become increasingly popular, and a growing number are equipped with wireless communications technology to increase their usefulness. However, this could make the devices susceptible to hackers.
Few IMDs encrypt signals, but this will soon change, said CTG’s Moyle.

Encryption could limit data interception and hide the commands used with the devices so that only permitted controllers could work with them.

However, noted Paul, there are limits to this approach because encryption capabilities could add complexity and require more system resources to function properly. Some IMDs might not have sufficient battery

Some experts have suggested implementing passwords that must be entered before someone can access an IMD.

However, doctors who might not know the password would have to be able to control the devices in case of emergency, particularly if the patient is unconscious. To deal with this, patients could wear bracelets that show their passwords. However, they could lose the bracelets.

One proposed solution popular with the security community—IMD-access passwords tattooed on patients as barcodes visible only under ultraviolet light—met with mixed results because some respondents didn’t like the idea of tattoos, explained Tamara
Topic 6
State-of-the-Art Multicore Computing

PARALLELISM VIA MULTITHREADED AND MULTICORE CPUs

Angela C. Sodan, Jacob Machina, Arash Deshmeh, Kevin Macnaughton, and Bryan Esbaugh, University of Windsor, Canada

IEEE Computer, March 2010
### Table 1. Comparison of features for current commercial multicore CPUs.

<table>
<thead>
<tr>
<th>Vendor</th>
<th>Product</th>
<th>Cores</th>
<th>Per core on-chip executing</th>
<th>Switching approach</th>
<th>Clock (GHz)</th>
<th>Power (watts per CPU)</th>
<th>Special features</th>
<th>On-chip Interconnect</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMD</td>
<td>Opteron (3rd generation)</td>
<td>4, 6</td>
<td>N/A</td>
<td>N/A</td>
<td>1.7-3.1</td>
<td>40-105</td>
<td>IMC, 128-bit FPU per core, dual PM*</td>
<td>Crossbar</td>
</tr>
<tr>
<td>AMD</td>
<td>Phenom II</td>
<td>3, 4</td>
<td>N/A</td>
<td>N/A</td>
<td>2.4-3.2</td>
<td>65-125</td>
<td>IMC, 128-bit FPU per core, dual PM*</td>
<td>Crossbar</td>
</tr>
<tr>
<td>AMD</td>
<td>Turion X2</td>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
<td>1.6-2.4</td>
<td>18-35</td>
<td>IMC</td>
<td>Crossbar</td>
</tr>
<tr>
<td>IBM</td>
<td>Core i7</td>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
<td>1.7-2.7</td>
<td>65</td>
<td>IR 4</td>
<td>Front-side bus**</td>
</tr>
<tr>
<td>IBM</td>
<td>Core i7</td>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
<td>1.8-3.3</td>
<td>65</td>
<td>IR 4</td>
<td>On-chip bus</td>
</tr>
<tr>
<td>AMD</td>
<td>Opteron (9000 series)</td>
<td>2</td>
<td>N/A</td>
<td>2/1</td>
<td>1.4-1.66</td>
<td>75-104</td>
<td>VLIW, IR 6</td>
<td>Direct pathways</td>
</tr>
<tr>
<td>Sun</td>
<td>UltraSPARC</td>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
<td>2.0-3.0</td>
<td>95-105</td>
<td>IR 4, dynamic PM</td>
<td>On-chip bus / front-side bus**</td>
</tr>
<tr>
<td>Sun</td>
<td>UltraSPARC</td>
<td>2</td>
<td>N/A</td>
<td>SMT</td>
<td>2.13-2.66</td>
<td>50-130</td>
<td>IR 4, dynamic PM</td>
<td>On-chip bus</td>
</tr>
<tr>
<td>IBM</td>
<td>Core 2 Quad</td>
<td>4</td>
<td>N/A</td>
<td>N/A</td>
<td>1.7-3.1</td>
<td>40-105</td>
<td>IMC, 128-bit FPU per core, dual PM*</td>
<td>Crossbar</td>
</tr>
<tr>
<td>IBM</td>
<td>Core 2 Duo Family</td>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
<td>1.7-2.7</td>
<td>65</td>
<td>IR 4</td>
<td>On-chip bus</td>
</tr>
<tr>
<td>Sun</td>
<td>UltraSPARC</td>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
<td>1.7-2.7</td>
<td>65</td>
<td>IR 4</td>
<td>On-chip bus</td>
</tr>
</tbody>
</table>

*IMC: Integrated Memory Controller

IEEE Computer, March 2010
Core complexity versus number of cores

Traditional CPU optimizations sought to increase the serial execution speed of a single thread, adopting techniques such as out-of-order execution, dynamic branch prediction, and longer pipelines for higher clock rates. The availability of thread-level parallelism in addition to instruction-level parallelism raises the major design decision of the extent necessary to simplify traditional CPU designs to allow the dedication of more circuitry to concurrency.

In regard to the decision between cores and hardware threads, for commodity computing the sweet spot seems to lie in hybrid designs. A small number of on-chip threads can be added for relatively little additional circuitry and can significantly increase throughput. However, diminishing returns in performance and increasing circuitry costs limit the gain from hardware threads. Thus, chip space beyond a few hardware threads is generally better exploited for more cores, cache, or other components. Hybrid CPUs have also been shown to be almost as energy efficient as pure multicore designs.
Rethinking Digital Design: Why Design Must Change

Power and cost issues necessitate rethinking digital design. To reduce design costs, we need to stop building chip instances, and start making chip generators instead. Domain-specific chip generators are templates that codify designer knowledge and design trade-offs to create different application-optimized chips.

IEEE Micro November/December 2010
The idea of specialization is well-known, and is already applied in varying degrees today. The use of single instruction, multiple data (SIMD) units (such as streaming SIMD extension [SSE], vector machines, or graphics processing units) as accelerators is an example of using special-purpose units to achieve higher performance and lower energy. To estimate how much potential gain is possible through specialization, we need only look at ASIC solutions, which often use orders of magnitude less power than general-purpose CPU-based solutions, while achieving the same or even greater performance levels.

ASICs are more efficient because they eliminate the overhead that comes with general-purpose computing. Many computing tasks, for example, need only simple 8- or 16-bit operations, which typically take far less than a picojoule in 90-nm technology. Efficiently executing these simple operations in a processor requires performing hundreds of operations per processor instruction, so the functional-unit energy becomes a significant fraction of the total energy.

Although their efficiency makes building customized chips preferable, designing them is expensive. The design and verification
Is a chip generator possible?

Although a chip generator would be a great tool to have, constructing a useful one is quite challenging. Here, we describe some of our early research on exploring ways to address this challenge. This early research falls into several areas. First, we want to quantitatively understand how specialization improves efficiency. Unless the energy...

Efficiency gains through customization

To better understand how customization can improve energy efficiency, we must first explicitly quantify the sources of overhead in general-purpose processors; then, we can explore methods to reduce them. As a case study, we examined H.264 video encoding (real-time 720p). H.264 has both highly data-parallel and sequential parts, making it an interesting application to study. In addition, there are both aggressive...
Reconfigurable Chip Multiprocessor as a Limited Generator

Figure A. The Stanford Smart Memories (SSM) chip multiprocessor architecture: a tile (1), a quad (2), and a system of quads (3). (Rx: receiver; Tx: transmitter.)