Activity Report 2011

Exploratory Action BYMOORE

Beyond Moore

RESEARCH CENTER
Saclay - Île-de-France

Table of contents

Exploratory Action BYMOORE

Keywords: Heterogeneous Multi-cores, Accelerators, Neural Networks, Defects, Energy

ByMoore is an exploratory action.

1. Members

Research Scientist

Olivier Temam [Team leader, Senior Researcher Inria, HdR]

Post-Doctoral Fellow

Zheng Li

2. Overall Objectives

2.1. Introduction

2.1.1. Summary

For several decades, it was possible to translate transistor size reduction into faster performing processors. Some additional architecture complexity was involved at each generation, but essentially, programs run on a given processor would run faster on the next-generation processor without modification. Due to power constraints, this evolution has considerably slowed down, leading to multi-core architectures, and raising severe programming (parallelization) challenges. Again due to power constraints, even the multi-core option is now being challenged. And at the same time, just as severe transistor fault issues are coming into play, calling for defect-tolerant architectures.

While the processor and multi-core options should still be pushed and optimized as much as possible, it is also now time to contemplate alternative computing systems designs that are more easily compatible with the power and defects issues in ultra-CMOS technology, as well as alternative silicon technologies or even non-silicon technologies. Companies can hardly afford to explore risky alternative paths, while academics could play an important pioneering and filtering role. Moreover, such computing approaches may induce profound enough changes in architecture and programming approaches that they should be investigated and anticipated far ahead.

We outline a roadmap that follows, adapts and attempts to take advantage of the current and upcoming technology options. Due to power reasons, we believe computing systems will have to shift to customization, making heterogeneous systems (composed of a mix of cores and accelerators) a prime citizen, including for general-purpose computing. As a first step, we feel it is necessary to craft the necessary hardware template and programming environment for such systems. We then want to focus on the design of accelerators; we contemplate fast methods for generating accelerators, but also configurable, and versatile (capable of targeting multiple algorithms) accelerators. We especially seek accelerators which can tolerate defects without even having to identify and disable faulty parts. That leads us to neural networks accelerators, which have inherent robustness properties; another possible contender, which will also be investigated later on, are accelerators based on probabilistic logic. While initially hyped, neural networks have fallen out of favor for several years, but a remarkable convergence of recent trends and innovations make them very interesting accelerator candidates again, with a broad application scope (in 2005, shift to RMS applications which rely on statistical and machine-learning algorithms), state-of-the-art algorithmic properties (in 2006, advent of Deep Networks) and emerging technologies almost ideally suited to their hardware implementation (in 2008, first implementation of a memristor device). We first explore robust digital CMOS implementations, then shift our focus to the more efficient and more inherently defect-tolerant analog CMOS implementations. Beyond CMOS, the memristor is a prime contender among emerging technologies for implementing neural network-based accelerators (and configurable accelerators as well). Beyond silicon, we also investigate other technologies, including hybrid silicon-biological implementations, and leverage recent progress in neurobiology which will allow to expand the application scope of these accelerators.

Overall, our driving goal is *to design computing systems that are low-power and defect-tolerant, which have a broad application scope and which can scale across many of upcoming and emerging technologies*.

2.1.2. Context

The observation at the root of this project is that we are entering a new era where technology constraints, and/or novel technologies, are already forcing us to consider, possibly drastically, different computing systems. The well-known Moore's law, stating that the size of transistors may be halved every two years or so, has been fulfilled for almost four decades. Like any exponential law, it is bound to stop; however many researchers have incorrectly predicted its end in the past few decades, and we will cautiously avoid betting against the ingenuity of physicists. However, while the transistor size is still regularly decreasing, we are now beyond the point where this regular decrease can be smoothly translated into processor performance improvements because of a set of technology-related constraints.

Power issues. Reducing transistor size has two main benefits: it allows to increase the transistor density (and then the chip capacity at a constant cost), and it allows to decrease the switching time of transistors (enabling higher clock frequencies). For several decades, processor manufacturers essentially leveraged the second property to increase processor performance, mostly taking advantage of the additional transistors to implement the mechanisms necessary to feed very fast cores with instructions and data fast enough. The first major roadblock was hit in 2005 when Intel announced it could no longer count on transistor size reduction to increase clock frequencies because of power dissipation constraints [10]. Each time a transistor switches, it dissipates tiny amounts of power; as the number of switches per second increases, the amount of power dissipated increases as well. It is possible to compensate for the power increase due to higher switching frequencies (and thus higher clock frequencies) by scaling down voltage. However, as transistor size and voltage are scaled down, another source of power dissipation, leakage power, increases and ultimately compensates for the benefit of voltage scaling. As a result, there is a crossing point, that we have reached, where voltage scaling is no longer useful for reducing the total power, and consequently, we can no longer afford to let transistors switch at maximum frequency. So, while transistor size can still be reduced, it is no longer possible to take advantage of their faster switching time due to excessive power dissipation. As a result, most processor manufacturers decided to leverage the increased transistor density only, not the faster transistor switching time, and turned to multi-core processors, instead of large processors with high clock frequency.

Because voltage is no longer scaled down and because the power requirement of a transistor does not decrease proportionally with its size, as we keep reducing the transistor size, the total power requirement of the same chip area (increasing number of transistors) actually increases. Because the power budget of a chip is limited, we are now facing a situation where it will not be possible to activate all transistors at the same time due to power limitations. These unused transistors have been coined *Dark Silicon* by ARM [17], the main embedded processor designer (used in all cell phones and many embedded systems). And they constitute a second major roadblock we are fast approaching. This roadblock may simply void the notion of many-cores: while transistor density allows for a large number of cores, it will no longer be possible to activate all cores at the same time.

Programming issues. In any case, the multi-core option was already severely challenged by programming issues: after several decades of research, there is still no easy solution for quickly and efficiently parallelizing a broad set of programs on a large number of cores.

Faults issues. While power issues are threatening the development of conventional architectures (fast cores, then multi-cores), it is still possible to scale down transistor size. However, as transistors get smaller, transistor faults are likely to raise even more challenges in the near future. With increasingly small transistors, it is no longer possible to ensure that all transistors and lines have the same characteristics. Such variations induce different latencies to hardware components, or different power characteristics. The manufacturers have to compensate for these variations either by over-designing or introducing targeted optimizations which further complicate designs.

More importantly, transistors are becoming so small that they become susceptible to transient faults and permanent defects. Transient faults can result in bit flips, where memory bits are inverted, for instance simply

due to cosmic rays. Permanent defects can either occur at design time, or even during the lifetime of the chip due to electromigration (the slow transfer of materials, i.e., chip aging, resulting in shorts and opens).

Overall, computing systems based on processors and multi-cores are severely challenged by the evolution of technology, which is no longer the smooth evolution enjoyed over the past four decades. Considering the regular improvement of processor performance has been the driver for a whole part of the economy, this issue has consequences way beyond computing systems.

2.1.3. Approach and Roadmap

2.1.3.1. Why seeking alternatives to processors ?

Because academia and industry are hugely familiar with and experienced in processor and multi-core design, one can expect significant progress can still be made by engineering solutions around power, defects and programming issues, and no one should discount how far this path can go. However, the pressure from technology is becoming so strong that seeking alternative paths should no longer be discounted either. Moreover, there are simple common sense arguments which further motivate the exploration of alternative paths. As mentioned before, transistor size reduction is bound to stop at some point. When that happens, or when the aforementioned constraints become too severe, computing systems will only keep improving by either resorting to different technologies, and/or different computing principles. The industry cannot afford to explore such risky paths, it is rather the role of academia to take such risks and filter out the most promising paths. Moreover, the transformations induced by alternative technologies or computing principles will require a long time before they mature enough to transfer to industry, so such research should be anticipated well ahead.

Naturally, there are many different possible alternative paths to be explored. In the paragraphs below, we outline the roadmap we intend to pursue, and the rationale for it.

2.1.3.2. Customization for low-power

Processors, also called, general-purpose processors, are flexible architectures: they can execute any algorithm. But this flexibility comes at a hefty power price: for performing a simple arithmetic or logic operation, a processor has to perform multiple steps, involving multiple power-hungry hardware blocks, in addition to the operation itself. In specialized circuits, also called accelerators or ASICs, there is no such overhead, only the operation is performed. Moreover, the size of specialized circuits is tailored to the task, which can drastically reduce power costs. Overall, custom circuits can perform the same task as processors with one or several orders of magnitude less power, and at the same, or sometimes, better performance. The caveat is naturally flexibility: by definition, a specialized circuit can perform only a single algorithm.

However, if it were possible to cram lots of different accelerators on one chip, then it would be possible to accelerate enough algorithms that most programs would be able to benefit. Now, the transistor density keeps increasing, but we know that only a fraction of transistors can be used simultaneously, i.e., the so-called dark silicon. However, using transistors for accelerator logic, i.e., trading cores for accelerator logic, is a convenient compromise. Unlike multi-cores which need to leverage many cores to speed up the execution of an algorithm, usually only one or a few accelerators are used to speed up the execution. Therefore, not all transistors need to be switched on at the same time, which makes them compatible with the notion of dark silicon.

Note that customization is not a new approach in any way. It has been commonplace in Systems-on-Chips (SoCs) in embedded systems for decades. What we are suggesting is to reconcile customization and flexibility and *use accelerators for general-purpose computing* by cramming enough and sufficiently flexible accelerators on the same chip.

The exact form of accelerators is open for debate. It can be either multiple specialized circuits, configurable logic, or intermediate solutions like versatile accelerators (useful for multiple, but not all, algorithms). In the case of specialized or configurable circuits, we need to find ways to streamline their design, by automatically generating circuits or automatically mapping high-level code on reconfigurable circuits. Also cores will always be needed for easily performing simple control tasks or for tasks not covered by accelerators, so we are contemplating heterogeneous systems composed of a mix of accelerators and cores, much like SoCs.

However, the key difference and the key pitfall we need to avoid is the *lack of programmability*, which currently hinders both multi-cores and SoCs; SoCs are notoriously difficult to program, especially to partition tasks among cores and accelerators. But progress in software engineering, especially *component-based programming*, are offering an elegant solution: for completely different reasons (programming productivity and managing large codes), software engineering practices have encouraged to decompose programs into strictly independent components. Each component tends to be a self-contained algorithm, and moreover, such programming practices are encouraging the reuse of algorithms across programs. Now, a component can either be executed as a software component on a programmable core, or replaced by a call to a hardware accelerator. As a result, a program decomposed into components (a painless task compared to program parallelization) could almost transparently take advantage of an architecture containing accelerators. This approach would simultaneously address the power, performance and programmability issues of multi-cores.

Main research steps:

- Focus on heterogeneous systems, composed of a mix of cores and accelerators to tackle power issues.
- Define a programming approach based on independent components that can be transparently mapped to software executing on cores or to calls to hardware accelerators.

2.1.3.3. Defect-Tolerant accelerators

Power issues motivated the switch from cores to accelerators (and heterogeneous multi-cores). Beyond power, we have explained that defects will likely become a dominant issue. Since accelerators will do most of the performance heavy lifting in the future, the challenge is now to *design defect-tolerant accelerators*.

A custom circuit, just like a core, is very vulnerable: a single faulty transistor might wreck it down. Configurable logic offers more defect-tolerance by creating functions out of many identical logic elements; if one element gets faulty, then the functions can be mapped to the remaining valid elements. However, this approach assumes it is possible to test and identify each individual logic element and shut it out if found faulty. While this approach is valid and should be explored, as the number of defects increases, the overhead of safely identifying and disabling faulty elements may significantly hurt scalability. As a result, we want to focus on approaches that keep operating correctly even in the presence of defects, and without having to identify and disable the faulty elements (be it transistors or more complex logic elements).

These constraints have led us to artificial neural networks, where information distribution and learning capabilities provide inherent robustness. Now, after the hype of the 1990s, where companies like Intel or Philips built commercial hardware systems based on neural networks, the approach quickly lost ground for multiple reasons: hardware neural networks were no match for software neural networks run on rapidly progressing general-purpose processors, their application scope was considered too limited, and even progress in machine-learning theory overshadowed neural networks.

However, in the past few years, a remarkable convergence of trends and innovations is casting a new light on neural networks and makes them very attractive candidate accelerators of future computing systems. With respect to scope, Intel outlined in 2006 [16] that the community was not focusing on the key emerging high-performance applications. It defined these key applications as *Recognition, Mining and Synthesis*, and coined the term RMS. Example applications are face recognition for security applications, data mining for financial analysis, image synthesis for gaming, etc. Now, many of these applications, especially in Recognition and Mining, rely on algorithms for which competitive versions exist based on neural networks. Even in the machine-learning community, the recent advent of Deep Networks [15], in 2006, has strongly revived interest in neural networks.

As a result, accelerators based on artificial neural networks have two key properties: they are inherently defect-tolerant and they are versatile accelerators, i.e., they can be used to tackle multiple core algorithms of several key RMS applications. That makes them ideal candidate accelerators if they can be architected to effectively sustain transistor defects.

Main research steps:

- Explore more conventional digital accelerators based on configurable logic where faulty elements can be identified and disabled.
- Define digital implementations of artificial neural networks (ANNs) which are robust to transistor defects.

2.1.3.4. Analog is inherently more defect-tolerant than digital

While digital ANNs can be made robust to defects, they remain an inefficient implementation: a fault on a low-order bit has little impact, but a fault on a high-order bit has a strong impact. In *analog* implementations, the magnitude of the value variation is correlated to the magnitude of the fault; so the effect of faults on the behavior of the circuit are more progressive. Analog has another asset: in embedded systems, the input is often originally analog (radio waves, sound, images,...), and an analog circuit would be able to process it natively, without digital conversion, saving circuit real-estate, power, and further improving robustness.

As a result, beyond digital accelerators, we want to investigate *analog accelerators*, especially analog neural network implementations. Neurons particularly shine for signal processing: because they are non-linear operators, they can easily implement complex analog functions such as integrators, which are commonplace in signal processing. As a result, complex signal processing tasks could be efficiently and directly implemented using neurons as operators, and learning could kick-in to compensate for errors if the function output deteriorates.

Main research steps:

- Investigate analog accelerators, especially based on neurons used either as analog operators, or as part of a neural network for learning-based compensation of errors.

2.1.3.5. Beyond CMOS, but still silicon ?

Both digital and analog implementations rely on *CMOS* transistors. At the core of our research is the notion that CMOS size reduction may stall at some point. We should thus start investigating what could be done beyond CMOS. Interestingly, some of the key contender alternatives to the CMOS transistor, still based on silicon, are mightily compatible with the approach developed so far.

For our purpose, a prime silicon-based contender alternative is the *memristor* [20]. Theorized by Chua in 1971 [13], the *memristor* is a novel silicon component which was effectively manufactured as a silicon device by Williams in 2008 for the first time [20]. This component implements a resistive memory: the resistance of the component can be changed and that resistance is memorized. This component is an almost ideal candidate for the implementation of either configurable logic (crossbars) or artificial synapses. In fact, the first memristor patents by Williams are about using the component to implement hardware artificial neural networks [19]. Synapses, which correspond to connections among neurons, memorize a weight applied to an input, i.e., almost the exact operating behavior of a memristor.

Among the other non-CMOS silicon contenders, PCMOS (Probabilistic CMOS) [12] is another interesting candidate. While this technology has been shown to be suitable to implement neural networks [11], it is also well suited for implementing probabilistic, also called randomized, algorithms.

The goal of PCMOS is to leverage the irregular behavior of transistors due to low voltage and process variation for computing purposes. The transistor is considered to provide the correct answer but only with a certain probability. By revisiting application algorithms so they are designed as *probabilistic* algorithms, it is possible to design whole circuits that take advantage of that property to conceive very low-power and defect-tolerant architectures. While PCMOS is not currently our primary focus, we will most likely investigate it for building a range of accelerator tiles.

Main research steps:

- Investigate hybrid implementations composed of CMOS logic (for neurons) and memristors (for synapses).

- Maintain a technology watch on other alternatives, such as PCMOS, and investigate related accelerators and their scope.

2.1.3.6. Beyond silicon ?

Beyond silicon, other alternative technologies are being contemplated, ranging from carbon nanotubes, graphene transistors to molecular-size transistors or quantum computing. It is actually quite possible that none of the alternative technologies will prevail and truly replace the transistor, but that they will simply coexist. It is all the more likely that each has particular strengths: as mentioned before, PCMOS is well suited for implementing probabilistic algorithms, memristor for implementing crossbars and synapses, quantum computing has strengths for certain categories of NP algorithms, etc. Whether technology unifies around a single approach or breaks down into multiple parallel paths, it remains compatible with the notion of heterogeneous systems and accelerators, where each accelerator would not only target certain algorithms, but would be designed with a certain technology. Consequently, this approach may actually largely shield us from the speculative nature of the upcoming technologies.

And among the possible technologies which are compatible with the approach developed so far in this document, i.e., the notion of neural network-based robust accelerators, biology emerges as a natural contender. While this may seem far-fetched at first sight, there already exist working implementations of transistors connected to individual biological neurons and forming information loops with observed transistor-to-neuron communications [14]. Infineon, an embedded systems company, has even developed a prototype chip, called the NeuroChip, for connecting a full layer of biological neurons with transistors.

Beyond biology as a technology, neurobiology may also bring a useful path for expanding the application scope of the contemplated accelerators. Already, detailed models, such as the HMAX model proposed by Poggio [18], show how to reconstruct sophisticated vision processing tasks using individual neurons, i.e., eligible for a replicated implementation in hardware solely using neurons. Whether they are implemented using silicon technology, or hybrid silicon-biological technology, these models would allow to significantly expand the nature of the tasks of these accelerators beyond what artificial neural networks can do. Moreover, as the understanding of complex neurobiological functions progresses, they could be leveraged for our accelerators.

Main research steps:

- Contemplate hybrid silicon-biology implementation of accelerators.
- Factor in progress in neurobiology to expand the application scope of accelerators beyond ANNs.
- Maintain a research watch on the progress of neurobiology to further expand application scope.

2.2. Highlights

- Zheng Li has obtained the Chinese government award for excellence in research in 2011, selecting the best PhDs conducting abroad by Chinese PhDs; in 2010, 506 Chinese PhDs in 29 countries and across all scientific disciplines have received the award.
- Olivier Temam has received a *Visiting Professorship for Senior International Scientists Award* from the Chinese Academy of Sciences for the cooperation with ICT, in 2011.

3. Software

3.1. Software

- **IODC:** Framework for implementing transparent iterative optimization in data centers, see Result 4.1.
- **P & R for neuromorphic accelerator:** A place and route software which maps a neural network graph on an analog neural network hardware, see Result 4.3.

- **HPT:** A performance comparison tool based on non-parametric tests, see Result 4.2.
- **Visual cortex model:** This model is initially based on Poggio's HMAX model, and has been reimplemented with the prospect of progressively moving it into hardware as efficiently as possible. This work is loosely connected to Result 4.4, and it is still work in progress.

4. New Results

4.1. Iterative Optimization for the Data Center (Alchemy-related research)

This result corresponds to research started within Alchemy, and it is less related to the objectives of ByMoore itself.

Iterative optimization is a simple but powerful approach that searches for the best possible combination of compiler optimizations for a given workload. However, each program, if not each data set, potentially favors a different combination. As a result, iterative optimization is plagued by several practical issues that prevent it from being widely used in practice: a large number of runs are required for finding the best combination; the process is inherently data set sensitive; and the exploration process incurs significant overhead that needs to be compensated for by performance benefits. Therefore, while iterative optimization has been shown to have significant performance potential, it is seldomly used in production compilers.

We propose [4] Iterative Optimization for the Data Center (IODC): we show that servers and data centers offer a context in which all of the above hurdles can be overcome. The basic idea is to spawn different combinations across workers and recollect performance statistics at the master, which then evolves to the optimum combination of compiler optimizations. IODC carefully manages costs and benefits, and is transparent to the end user.

We evaluate IODC using both MapReduce and throughput server applications. In order to reflect the large number of users interacting with the system, we gather a very large collection of data sets (at least 1000 and up to several million unique data sets per program), for a total storage of 10.7TB, and 568 days of CPU time. We report an average performance improvement of 1.48×, and up to 2.08×, for the MapReduce applications, and 1.14×, and up to 1.39×, for the throughput server applications.

4.2. Statistical Performance Comparisons of Computers (Alchemy-related research)

This result corresponds to research started within Alchemy, and it is less related to the objectives of ByMoore itself.

As a fundamental task in computer architecture research, performance comparison has been continuously hampered by the variability of computer performance. In traditional performance comparisons, the impact of performance variability is usually ignored (i.e., the means of performance measurements are compared regardless of the variability), or in the few cases where it is factored in using parametric confidence techniques, the confidence is either erroneously computed based on the distribution of performance measurements, instead of the distribution of sample mean of performance measurem ents, or too few measurements are considered for the distribution of sample mean to be normal. We first illustrate how such erroneous practices can lead to incorrect comparisons.

Then, we propose [3] a non-parametric Hierarchical Performance Testing (HPT) framework for performance comparison, which is significantly more practical than standard p arametric confidence tests because it does not require to collect a large number of measurements in order to achieve a normal distribution of the sample mean. This HPT framework has been implemented as an open-source software.

4.3. Implementation of Signal Processing Tasks on Neuromorphic Hardware

Because of power and reliability issues, computer architects are forced to explore new types of architectures, such as heterogeneous systems embedding hardware accelerators. Neuromorphic systems are good candidate accelerators that can perform efficient and robust computing for certain classes of applications. We propose[9] a spiking neurons based accelerator, with its hardware and software, that can be easily programmed to execute a wide range of signal processing applications. A library of operators is built to facilitate implementation of various types of applications. Automated placement and routing software tools are used to map these applications onto the hardware. Altogether, this system aims at providing to the user a simple way to implement signal processing tasks on neuromorphic hardware.

4.4. Automatic Abstraction and Fault Tolerance in Cortical Microachitectures

Recent advances in the neuroscientific understanding of the brain are bringing about a tantalizing opportunity for building synthetic machines that perform computation in ways that differ radically from traditional Von Neumann machines. These brain-like architectures, which are premised on our understanding of how the human neocortex computes, are highly fault-tolerant, averaging results over large numbers of potentially faulty components, yet manage to solve very difficult problems more reliably than traditional algorithms. A key principle of operation for these architectures is that of automatic abstraction: independent features are extracted from highly disordered inputs and are used to create abstract invariant representations of the external entities. This feature extraction is applied hierarchically, leading to increasing levels of abstraction at higher levels in the hierarchy. This work [6] describes and evaluates a biologically plausible computational model for this process, and highlights the inherent fault tolerance of the biologically-inspired algorithm. We introduce a stuck-at fault model for such cortical networks, and describe how this model maps to hardware faults that can occur on commodity GPGPU cores that used to realize the model in software. We show experimentally that the model software implementation can intrinsically preserve its functionality in the presence of faulty hardware, without requiring any reprogramming or recompilation. This model is a first step towards developing a comprehensive and biologically-plausible understanding of the computational algorithms and microarchitecture of computing systems that mimic the human cortex, and to applying them to the robust implementation of computational tasks on future computing systems built of faulty components.

5. Contracts and Grants with Industry

5.1. Grants with Industry

- **ANR MHANN** (Memristive Hardware Artificial Neural Networks Accelerators): The purpose of this project is to build a medium scale prototype of such a bio inspired architecture, by using long life and nanometric-ferroelectric memristors. The area, performance and power benefits of this approach will be evaluated to define its interest for embedded systems. The MHANN project is multi disciplinary in the sense that it proposes new physical concepts for devices (physics) and aims at integrating them into on chip bio inspired architectures (micro electronics, computer science and architectures).
- **ANR NEMESIS** (NEuroMorphic hardwarE for Smart vIsion Sensor): This project aims at exploring the potential of biologically-inspired spike-based image processing supported by the realization of massively parallel yet scalable hardware thanks to 3D stacking of integrated circuits.
- **ANR Arch2Neu** (Neuromorphic hardware and software environment for versatile computing): Arch2Neu aims at investigating the potential of neuromorphic architectures for computing purposes, and particularly for signal-processing applications. We develop analog neural hardware, interconnections architectures, libraries, and compilers to provide to the user a versatile and efficient computing machine. You can learn more about our research through the dedicated webpages.

6. Partnerships and Cooperations

6.1. European Initiatives

6.1.1. FP7

- **European Network of Excellence HiPEAC2 and HiPEAC3:** HiPEAC is a network of excellence on High-Performance Embedded Architectures and Compilers. It involves more than 70 European researchers from 10 countries and 6 companies, including ST, Infineon and ARM. The goal of HiPEAC is to steer European research on future processor architectures and compilers to key issues, relevant to the European embedded industry.

6.2. International Initiatives

6.2.1. INRIA Associate Teams

- **YOUHUA**: ICT-INRIA associate team. The goal of the team is to investigate a programming approach for heterogeneous multi-cores.

  The likely path forward for architectures are heterogeneous multi-cores composed of a mix of cores and hardware accelerators (ASICs or reconfigurable circuits). Now, whether the architectures are homogeneous multi-cores or heterogeneous multi-cores, the difficulty to efficiently program such architectures remains the key issue. We propose a programming approach that is pragmatic and capable of letting non-expert users take advantage of the performance of homogeneous and heterogeneous multi-cores. Rather than asking programmers to understand architectures and write parallel or RTL (for accelerators) versions of their code, we ask programmers to explicit the algorithms they are using within their codes, and we rely on expert programmers to provide efficient parallel or RTL implementations of these algorithms. Not only this approach can make it possible for non-expert users to take advantage of complex architectures, but it also makes programs portable across a broad range of architectures, and furthermore, it considerably expands the opportunities for automatically tuning applications and architectures.

6.2.2. Visits of International Scientists

- Jing Huang sent by ICT and Chinese Academy of Science for 10 months in France, for cooperation on reconfigurable accelerator.
- Numerous stays in China in 2011 by Olivier Temam (about once per month).

6.2.3. Participation In International Programs

- **YOUHUA at LIAMA:** LIAMA is (originally) an INRIA-Chinese Academy of Sciences lab (now Europe-China CS lab), and we just established a joint team at LIAMA, also called YOUHUA. Unlike YOUHUA, this joint team is INRIA-ICT-EPFL. The goal is both the design of reconfigurable accelerators, and programming approaches for heterogeneous multi-cores.

7. Dissemination

7.1. Animation of the scientific community

- General Chair of the International Symposium on Code Generation and Optimization (CGO), 2011.
- Program Chair of the International Conference on High-Performance and Embedded Architectures (HiPEAC), 2011.

- Associate Editor of IEEE Micro.
- Program Committees: ISCA 2011, ISCA 2012, and several workshops.

7.2. Teaching

Master : Architecture course at Ecole Polytechnique (about 35 hours per year), France.

PhD : Taj Khan, Robust sampling techniques for speeding up processor simulation, University of Paris Sud, June 2011, Olivier Temam and Daniel Gracia-Perez.

8. Bibliography

Major publications by the team in recent years

[1] M. BREUGHE, Z. LI, Y. CHEN, S. EYERMAN, O. TEMAM, C. WU, L. EECKHOUT. *How Sensitive is Processor Customization to the Workload's Input Datasets?*, in "IEEE, International Symposium on Application Specific Processors (SASP), in conjunction with", San Diego, CA, June 2011.

[2] O. CERTNER, Z. LI, A. RAMAN, O. TEMAM. *A Very Fast Simulator for Exploring the Many-Core Future*, in "IEEE, International Parallel & Distributed Processing Symposium (IPDPS), in conjunction with", Anchorage, Alaska, May 2011.

[3] T. CHEN, Y. CHEN, Q. GUO, O. TEMAM, Y. WU, W. HU. *Statistical Performance Comparisons of Computers*, in "IEEE, International Symposium on High-Performance Computer Architecture (HPCA), in conjunction with", New Orleans, Louisiana, February 2012.

[4] Y. CHEN, S. FANG, L. EECKHOUT, O. TEMAM, C. WU. *Iterative Optimization for the Data Center*, in "ACM/IEEE, International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), in conjunction with", London, UK, March 2012.

[5] H. ESMAEILZADEH, S. GIRBAL, K. MCKINLEY, O. TEMAM, S. YEHIA. *Programming Heterogeneous Hardware Components with Software Components*, in "Workshop on SoC Architecture, Accelerators and Workloads (SAW), in conjunction with International Symposium on High-Performance Computer Architecture (HPCA)", San Antonio, Texas, February 2011.

[6] A. HASHMI, H. BERRY, O. TEMAM, M. LIPASTI. *Automatic Abstraction and Fault Tolerance in Cortical Microarchitectures*, in "ACM/IEEE, International Symposium on Computer Architecture (ISCA), in conjunction with", San Jose, CA, June 2011.

[7] R. HELIOT, A. JOUBERT, O. TEMAM. *Robust and Low-Power Accelerators based on Spiking Neurons for Signal Processing Applications*, in "International Workshop on Design for Reliability (DFR), in conjunction with International Conference on High Performance Embedded Architectures and Compilers (HiPEAC)", Heraklion, Greece, January 2011.

[8] T. KHAN, D. GRACIA-PEREZ, O. TEMAM. *Combining On-Line Sampling and Adaptive Warm-Up for a More Practical Sampling Strategy*, in "International Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools (RAPIDO), in conjunction with International Conference on High Performance Embedded Architectures and Compilers (HiPEAC)", Heraklion, Greece, January 2011.

[9] O. TEMAM, R. HELIOT. *Implementation of Signal Processing Tasks on Neuromorphic Hardware*, in "IEEE, International Joint Conference on Neural Networks (IJCNN), in conjunction with", San Jose, CA, June 2011.

References in notes

[10] S. Y. BORKAR, P. DUBEY, K. C. KAHN, D. J. KUCK, H. MULDER, E. R. M. RAMANATHAN, V. THOMAS, I. CORPORATION, S. S. PAWLOWSKI. *Intel ® Processor and Platform Evolution for the Next Decade Executive Summary*, 2006.

[11] L. N. CHAKRAPANI, B. E. S. AKGUL, S. CHEEMALAVAGU, P. KORKMAZ, K. V. PALEM, B. SESHASAYEE. *Ultra-efficient (embedded) SOC architectures based on probabilistic CMOS (PCMOS) technology*, in "Design, Automation, and Test in Europe", Munich, 2006, 1110.

[12] S. CHEEMALAVAGU, P. KORKMAZ, K. V. PALEM. *Ultra low-energy computing via probabilistic algorithms and devices: CMOS device primitives and the energy-probability relationship*, in "International Conference on Solid State Devices", Tokyo, 2004, p. 2–4.

[13] L. CHUA. *Memristor-The missing circuit element*, in "IEEE Transactions on Circuit Theory", 1971, vol. 18, n^{o} 5, p. 507–519 [*DOI :* 10.1109/TCT.1971.1083337].

[14] P. FROMHERZ, A. STETT. *Silicon-Neuron Junction: Capacitive Stimulation of an Individual Neuron on a Silicon Chip*, in "Physical Review Letters", August 1995, vol. 75, n^{o} 8, p. 1670–1673 [*DOI :* 10.1103/PHYSREVLETT.75.1670].

[15] H. LAROCHELLE, D. ERHAN, A. COURVILLE, J. BERGSTRA, Y. BENGIO. *An empirical evaluation of deep architectures on problems with many factors of variation*, in "International Conference on Machine Learning", New York, New York, USA, ACM Press, 2007, p. 473–480 [*DOI :* 10.1145/1273496.1273556].

[16] B. LIANG, P. DUBEY. *Recognition, Mining and Synthesis*, in "Intel Technology Journal", 2005, vol. 09, n^{o} 02 [*DOI :* 10.1535/ITJ.0902.F].

[17] M. MULLER. *Dark Silicon and the Internet*, in "EE Times "Designing with ARM" virtual conference", 2010.

[18] T. SERRE, L. WOLF, S. BILESCHI, M. RIESENHUBER, T. POGGIO. *Robust object recognition with cortex-like mechanisms.*, in "IEEE transactions on pattern analysis and machine intelligence", March 2007, vol. 29, n^{o} 3, p. 411–26 [*DOI :* 10.1109/TPAMI.2007.56].

[19] G. SNIDER. *Molecular-junction-nanowire-crossbar ... - Google Patent Search*, 2008.

[20] R. S. WILLIAMS. *How We Found the Missing Memristor*, in "IEEE Spectrum", 2008.