SEVENTH FRAMEWORK PROGRAMME

HiPEAC COMPILATION ARCHITECTURE

High Performance and Embedded Architecture and Compilation

Computing Systems: Research Challenges Ahead The HiPEAC Vision 2011/2012

M. Duranton, D. Black-Schaffer, S. Yehia, K. De Bosschere

This document was produced as a deliverable of the FP7 HiPEAC Network of Excellence under grant agreeement ICT-217068.
October 2011

Contents

Executive Summary

Computing Systems have a tremendous impact on everyday life in all domains, from the Internet to consumer electronics, transportation to manufacturing, medicine, energy, and scientific computing. In the future, computing systems will continue to be one of our most powerful tools for taking on the societal challenges shaping Europe, its values, and its global competitiveness.

The FP7 HiPEAC network of excellence is Europe's premier organization for coordinating research, improving mobility, and enhancing visibility in the computing system field. HiPEAC covers all computing market segments: embedded systems, general purpose computing systems, data centers and high performance computing. Created in 2004, HiPEAC today gathers over 250 leading European academic and industrial computing system researchers from about 100 universities and 50 companies in one virtual centre of excellence. To encourage computing systems innovation in Europe, HiPEAC provides collaboration grants, internships, sabbaticals, and improves networking through the yearly HiPEAC conference, ACACES summer school, and the semiannual computing systems week.

In this roadmap document, HiPEAC leverages the broad expertise of its members to identify and analyze the key challenges for computing systems in Europe over the next decade. While advances in computing systems have been consistent and dramatic over the past fifty years, its future today is not as certain. To continue to be a tool for providing new and innovative solutions, the computing systems community must face serious challenges in *efficiency, complexity,* and *dependability*.

Definite trends are emerging from upcoming societal challenges and the evolution of computing systems. First, our society is clearly experiencing a new *era of data* explosion in all domains. This explosion of data is particularly characterized by the variety of formats data can take (text, documents, video, photos, environment observations, etc.). Second, while connectivity during the last decade was mainly limited to wired computers and servers, we are now witnessing an *explosion in connectivity.* Critically, this connectivity now comprises a large variety of devices, ranging from warehouse-sized data centers for cloud and high performance computing, to mobile devices (phones, cars, planes, etc.), and all the way down to embedded sensors in the physical world and in the human body. Third, the computing domain is facing an increased *demand for dependability* and reliability across all fields. Many emerging applications require high levels of safety and security (healthcare, automotive, etc.) and new technologies are introducing new challenges in reliability (ubiquitous connectivity, unreliable devices, etc.).

On the market side, the leadership of the PC as driver for hardware and software development is fading, and being replaced by more mobile and consumer-oriented devices. Accordingly, the focus on development is shifting to embedded/mobile systems and cloud services. This transition is leading to major *de-verticalization* of the market players and a *convergence* of technology platforms. As a result, we are experiencing an increased diversification of the value chain and more emphasis on integration. While this encourages entry to the market, it makes product differentiation more difficult.

From a technology point of view, the "Moore's law" of ever-increasing levels of integration fuelled performance over the past five decades. Each new technology generation doubled transistor density and increased frequency, while simultaneously reducing the power per transistor. Ever more demanding applications directly exploited these growing resources with minimal changes to the software.

However, a major paradigm shift is now taking place:

1) "Moore's law", while keeping pace in terms of transistor density, is now enabling only minor frequency increases and minor decreases in power dissipation per transistor. To keep increasing raw performance, the current approach is to add more processing units (multi-core processing). Unfortunately, this is far from transparent to most applications: existing software now has to be re-engineered to execute efficiently on parallel architectures. The *complexity* of this task is one of today's main challenges.
2) Another important limitation is *power efficiency*: even if more devices can be packed on a chip, the power used by each device is no longer dropping accordingly (end of Dennard scaling). Since we are already at the power limit, it will no longer be possible to use all devices on a chip simultaneously. The resulting need to turn off functionality to meet power constraints results in "Dark Silicon".
3) The *explosion of data* (the "Data Deluge") and the increase in natural (unstructured) data from the real world ("cyber-physical systems") is increasing computation requirements, and demanding new computing methods and storage faster than technology can keep up. Increasingly complex algorithms and systems are required to efficiently handle this new era of data.
4) As devices become smaller with each generation, the variability between devices (in terms of performance and power) and their reliability decreases. To continue to leverage ever-smaller devices, we must learn how to build reliable systems from unreliable, and highly variable, components.

For the short and medium term, HiPEAC believes that specializing computing devices is the most promising path for dramatically improving *power efficiency.* This improved efficiency is needed to meet the *data deluge* of the 21st century. Unfortunately this trend will only worsen the *complexity* and cost of developing software for these systems. Further, the increasing need for *reliability* forces us to consider variability, security, and safety at all levels of the system and development cycle. In this light, HiPEAC has identified seven specific research objectives for the computing systems community:

Efficiency (with a focus on energy efficiency)

1) ***Heterogeneous computing systems:*** how can we design computer systems to maximize power efficiency and performance?
2) ***Locality and communications management:*** how do we intelligently minimize or control the movement of data to maximize power efficiency and performance?

System Complexity

3) ***Cost-effective software for heterogeneous multi-cores:*** how do we build tools and systems to enable developers to efficiently write software for future heterogeneous and parallel systems?
4) ***Cross-component/cross-layer optimization for design integration:*** how do we take advantage of the trend towards component-based design without losing the benefits of cross-component optimization?
5) ***Next-generation processor cores:*** how do we design processor cores for energy-efficiency, reliability, and predictability?

Dependability and applications (with a focus on their non-functional requirements)

6) ***Architectures for the Data Deluge:*** how can we tackle the growing gap between the growth of data and processing power?
7) ***Reliable systems for Ubiquitous Computing:*** how do we guarantee safety, predictability, availability, and privacy for ubiquitous systems?

In the longer term, it will become critical to investigate research directions breaking with the line of classical Von Neumann systems and the traditional hardware/software boundary. This includes new devices, such as dense non-volatile memories, optical interconnect, spintronics, memristors, etc., and new computing paradigms, such as bio-inspired systems, stochastic computing, swarm computing, etc. These directions all offer the promise of performing particular tasks at high efficiency levels while decreasing the impact of the constraints of the new technology nodes.

By addressing the seven specific research objectives and investigating emerging technologies, we will be able to ensure that Europe can continue to benefit from the promised growth of computing systems technology. Failure to address these challenges will significantly reduce our ability to leverage computing systems's potential to improve global competitiveness and tackle society's challenges.

Computing Systems: Research Challenges Ahead The HiPEAC Vision 2011/2012

1. Computing Systems: The Cornerstone of Our Civilization

Computing systems devices are universal today. All facets of public, private, and commercial life are impacted both directly and indirectly by them. Advances in computing systems are the key to the development of new domains and revolutionary technologies, such as personalized medicine, online social interaction, and immersive entertainment experiences. Indeed, computing systems are so valuable that people demand constant access and have an insatiable appetite for new devices and capabilities. In addition to creating new paradigms, computing capabilities revolutionize existing technologies. Across all of modern society, from manufacturing to agriculture, communications to energy, and social interaction to advanced science, computing systems is our primary tool for improving productivity, safety, well-being, and health. Investing in computing systems strengthens our most powerful tool for tackling the problems of today and tomorrow.

Yet today computing systems are experiencing several dramatic shifts. Technological limitations are pushing computing systems away from the ever-increasing performance of the past, while applications are pulling computing systems towards ever larger, more intensive, and more critical roles. At the same time business trends are causing widespread convergence of platforms, decoupling of design and production, and a rapid switch towards mobile embedded systems over desktop computers.

1.1. Technology Push

A decade into the 21st century, **computing systems are facing a once-in-a-lifetime technical challenge: the relentless increases in raw processor speed and decreases in energy consumption of the past 50 years have come to an end.** As a result, all of computing systems are being forced to switch from a focus on performance-centric serial computation to *energy-efficient parallel* computation. This switch is driven by the higher energy-efficiency of using many slower parallel processors instead of a single high-speed one. However, existing software is not written to take advantage of parallel processors. To benefit from new processor developments, developers must re-design and re-write large parts of their applications at astronomical cost.

Yet even the shift to *universal parallelism* is not enough. The increasing number of components on a chip, combined with decreasing energy scaling, is leading to the phenomenon of *"dark silicon"*, whereby chips have a too high power density to use all components at once. This puts an even greater emphasis on efficiency, and is driving chips to use multiple different components, each carefully optimized to efficiently execute a particular type of task. This era of *heterogeneous parallel computing* presents an even greater challenge for developers. Now they must not only develop parallel applications, but they are responsible for deciding what types of processors to use for which calculations.

Tackling these challenges requires addressing both the hardware and software challenge. We must design energy-efficient systems with the right mix of heterogeneous parallel components and provide developers with the tools to effectively leverage them. Without either developments, we will be unable to continue the computing growth that has so changed our society over the past 50 years. Accomplishing this will require a global reassessment of how hardware and software interact.

Critical Trends Influencing Computing Systems Today

Applications	Technology	Business
• Data Deluge • Intelligent Processing • Ubiquitous Communication	• Frequency Limits • Power Limits • Dark Silicon	• Convergence • Specialization • Post-PC Devices

1.2. Application Pull

While technology is pushing computing systems towards heterogeneous parallelism for energy efficiency, applications are pulling it towards ever increasing levels of performance, connectivity, and dependability. Individuals, businesses, governments, scientists and societies alike are relying on cost-effective, robust, and ubiquitous storage, communication and processing of unprecedented volumes of data. At the same time, the demand for more *intelligent processing* is growing, largely due to the increasingly unstructured nature of the data that is increasingly provided by the physical world. The resulting "*Data Deluge*" is far out-pacing any projected advances in storage and processing capacity.

While we are struggling to cope with storing and processing the on-going data deluge, the modalities for using the information have changed dramatically. Users are exploiting *ubiquitous communications* to change where and how computing is done. As a result, backend processing is moving from fixed-purpose servers to general-purpose, commodity, cloud systems, and user interaction is shifting from the desktop to the embedded devices, such as smartphones and tablets. This transition enables more flexible and scalable computing, but puts a much heavier emphasis on dependability and security. As cloud and communications systems become integral to all aspects of daily life, we become dependent on them for safety-critical functions and we rely on them to protect our privacy and security. To survive this transition we need to develop techniques for building large distributed systems that can meet society's dependability, security, and privacy requirements.

The combination of massive amounts of data, demand for intelligent processing, ubiquitous communication, and constrained system energy efficiency, lead us to summarize the trend in applications as: ***"Data Deluge meets the Energy Wall in a Connected World."*** To meet the challenges posed by these trends **we need to enable storage, communications, and processing with orders of magnitude less energy than we can today, while ensuring functional dependability and information security.** Accomplishing these goals requires revisiting the design of applications and the systems upon which they are built.

1.3. Business Trends

The computing systems business is likewise experiencing a range of disruptive trends. *Convergence*, both in hardware and software platforms, is rampant throughout the industry with desktop processors and embedded processors merging and applications moving from local systems to commodity cloud platforms and the web. Consumers are discovering that *"less is more"* and are seeking improved mobility and experience over raw performance and features. Companies are *de-verticalizing* and spinning off parts of the value chain to improve competitiveness by increasing specialization and productivity at each level. And at the same time, the move towards open source software has opened up new *collaborations* across companies and nations, and ushered in a vast range of robust, low-cost tools and technologies.

These trends are putting increased pressure on companies to effectively *integrate* software and hardware components. De-verticalization means designers no longer control the whole value chain, and must combine components from a range of suppliers. Convergence and the "less is more" trend are forcing companies to compete on the whole product package and ecosystem, rather than the raw performance and feature list. And the availability of open source software has both lowered the barrier for entry and increased the need for product differentiation. These trends are all shifting the market from the historic leadership of computing on the desktop to a new focus on *mobile devices accessing commodity cloud systems.* To be competitive in this market, companies must either excel at integrating components and systems from diverse manufacturers and delivering an optimized end-user experience, or take the opposite approach and control every level of the value chain (e.g. Apple and Google).

2. The HiPEAC Core Computing Systems Challenges

Several aspects of the future of computing systems for the next several years are clear:

- **Energy efficiency** will force hardware to move to heterogeneous parallel systems
- The **Data Deluge** will drive applications towards increasing levels of real time processing of increasingly sophisticated data
- Ubiquitous computing and "less is more" will force a business focus away from the desktop towards the **cloud and mobile devices**

Yet these same trends lead to significant challenges:

- Heterogeneous systems are prohibitively difficult (and hence costly) to program with today's tools
- Existing infrastructures for data processing will not scale up to meet the expected increase in data
- The focus on mobile devices and cloud processing will result in significant challenges for providing reliable services

Based on these trends and challenges, HiPEAC has identified three Core Computing Systems Challenges:

The HiPEAC Core Computing Systems Challenges

- ***Efficiency:*** Efficiency focuses on maximizing the amount of computation we can accomplish per unit of energy and for a minimum cost (both development and production), and is the key for sustaining growth in our computational capabilities.
- ***Complexity:*** Complexity identifies the need to provide tools and techniques for enabling developers of software and new hardware to leverage increasingly complex systems for increasingly complex applications.
- ***Dependability:*** Dependability encompasses the reliability and predictability needed for safety-critical systems and the security and privacy demanded for ubiquitous computing.

Each of these challenges plays an integral role for the future growth of our computing capabilities and the societal benefits we derive from them.

Core Computing Systems Challenges

Efficiency	Complexity	Dependability
• Power • Performance	• Parallelism • Heterogeneity	• Reliability • Privacy

2.1. Efficiency

Power defines performance for all modern and future computing systems. From battery life in mobile devices to cooling capacity in large-scale data centers, the key metrics of computing systems are now *Operations/Watt* and *Operations/Watt/Euro*. Performance at any cost is no longer tenable. The future is in efficiency first, and as a result, it is essential to optimize energy usage throughout the system.

The solution to improved energy efficiency is to leverage *parallel heterogeneous architectures* of task-optimized processors and accelerators. By optimizing these components for specific tasks, their energy efficiency can be increased by orders of magnitude. However, specialization comes with a loss of generality. As a result, there will be a significant burden on system designers and application developers to choose the right combination of heterogeneous processors and accelerators, and to leverage them optimally in the applications.

2.2. Complexity

Complexity has a strong impact on the cost of developing computing systems. As systems and applications become more complex and distributed, the difficulties in design, implementation, verification, and maintenance are rising. The issue of complexity has come to the forefront with the move to *universal parallelism,* which is widely acknowledged as being too complex to expose to developers. Add to this the further *complication of heterogeneity,* and the resulting complexity becomes fatal for innovation and advancement. As a result, it is no longer practical to write software that fully leverages modern and future systems.

The solution to this increased complexity is to develop tools and techniques that handle the complexity and simplify the development for system designers and application developers. These must span the full range from design space exploration for hardware and performance modeling for software, to runtime analysis, virtualization, optimization, debugging, and high-level programming systems. The goal is to provide a *simplified interface* for developing and understanding applications, a guarantee of *performance portability* across current and future systems, and a path for *integrating legacy code*. Without these capabilities, the costs of leveraging modern and future hardware will be too high, and the societal advances enabled by computing systems will stall.

2.3. Dependability

Dependability defines the safety, security, and reliability of computing systems. **All safety-critical systems today are based on computing systems technology, and with the promise of increased performance, connectivity, and reduced size, such systems will play an increasing role in the future.** In addition to safety-critical systems, the global accessibility of data is bringing issues of data *privacy and security* to the forefront. For society to benefit from the massive amounts of data available we need to ensure that individual privacy and data ownership can be respected and enforced.

But dependability is not just about the design and construction of secure and reliable systems. As technology advances, the individual devices from which systems are built are becoming less and less reliable themselves, and systems must adapt. To counter this, we must develop techniques for building systems from *unreliable components* without unduly sacrificing performance, efficiency, or cost.

The solution to handling increased demands for dependability must be built into all layers of the system. At the hardware layer improved predictability and security must be part of the basic architecture, while the software stack must include time and latency as first-class requirements. Tools must provide analysis and verification to guarantee correctness and improved statistical timing models to predict system behavior. In addition, systems must work together with their hardware to adapt to failing and unreliable components, while still maintaining the required level of dependability.

3. Impact of Computing Systems on Society

We must make significant advances in all three Core Challenges to maintain the fantastic growth rates that have made computing the cornerstone of our modern civilization. If we fail in any one of them, we will risk the future advances promised by more powerful, ubiquitous, and efficient computation. To highlight the importance of these challenges for society, the table below identifies key applications and how they relate to the nine societal grand challenges as identified by the commission [ISTAG].

Efficiency	Complexity	Dependability
Energy		
Reduce the direct energy consumption of computing systems. High-performance for optimizing energy usage, generation, and distribution.	Large-scale distributed power monitoring and generation networks. (e.g., smart meters).	Safety and reliability for generation and distribution. Privacy for personal energy consumption information while enabling aggregate analysis.
Transportation and Mobility		
Reduced power consumption for smarter vehicles and sensors. High-performance for design, optimization of routing and planning.	Large-scale networks of cars and smart roads. Optimization of goods delivery and transportation.	Safety of embedded vehicle systems. Reliability of global transportation optimizations. Privacy for personal location data, while enabling aggregate analysis.
Healthcare		
Reduced power consumption for smarter and smaller sensors and diagnostic tools. High-performance for drug design and population analysis.	Large-scale systems for medical record analysis and patient monitoring.	Safety of embedded medical devices. Privacy for personal medical data while enabling aggregate analysis.
Aging Population		
Reduced power consumption for smarter home sensors and household robotics.	Large-scale systems for home monitoring. Complex robotics for human interaction.	Safety of embedded medical devices and household services. Privacy for personal data and monitoring.
Environment		
Reduced power consumption for smarter and smaller sensors. High-performance for global-scale simulation, analysis and visualization of data.	Large-scale systems for integrating data from networks of sensors.	Reliable monitoring of critical environmental markers. Privacy for personal data while enabling aggregate analysis.
Productivity		
Reduced power for portable embedded systems. High-performance for product optimization, forecasting, and efficient manufacturing.	Large-scale, real-time integration of data from manufacturing, distribution, and sales to enable optimized production.	Optimizing high-reliability with low cost, particularly in the presence of unreliable components.

	Efficiency	Complexity	Dependability
Safety	Reduced power for smaller, more intelligent embedded systems. High-performance for more intelligent analysis of complex situations.	Verifying integration of components from multiple vendors.	Safety and reliability of embedded devices and safety critical systems.
Security	Smaller, higher-performance tools for law-enforcement and defense.	Large-scale, real-time data analysis for detecting threats and patterns.	Security and privacy guarantees for individuals and data.
Education	Reduced power for smaller systems to provide ubiquitous access to information. High-performance for more powerful and intuitive learning tools.	Enabling non-computing professionals to leverage computing advances through higher-level tools.	Protection from unappropriate content. Guarantees for secure and safe operation.

4. HiPEAC Research Objectives in the European Context

To address these challenges, HiPEAC has identified three key areas for research (efficiency, system complexity, and applications) and seven specific research objectives:

- **Efficiency (with a focus on energy efficiency)**
  - **Heterogeneous computing systems:** how can we design computer systems to maximize power efficiency and performance?
  - **Locality and communications management:** how do we intelligently minimize or control the movement of data to maximize power efficiency and performance?
- **System Complexity**
  - **Cost-effective software for heterogeneous multi-cores:** how do we build tools and systems to enable developers to efficiently write software for future heterogeneous systems?
  - **Cross-component/cross-layer optimization for design integration:** how do we take advantage of the trend towards component-based design without losing the benefits of cross-component optimization?
  - **Next-generation processor cores:** how do we design processor cores for energy-efficiency, reliability, and predictability?
- **Applications (with a focus on their non-functional requirements)**
  - Architectures for the Data Deluge: how can we tackle the growing gap between the growth of data and processing power?
  - Reliable systems for Ubiquitous Computing: how do we guarantee safety, availability, and privacy for ubiquitous systems?

By focusing on these areas, the HiPEAC community will be able to make significant high-impact contributions to computing in Europe and in the world. These advances are necessary to enable our society in the 21st century to continue to reap the benefits of computing systems that have so revolutionized the 20th century.

4.1. Efficiency

4.1.1. Heterogeneous computing systems

Drive

The end of power scaling combined with continued increases in transistor density have put computing systems in the difficult position of having more transistors than can be turned on at once. This era of "dark silicon" leads to a focus on making the most efficient use of the transistors that are turned on at any given time. As a result, processor design is becoming heterogeneous, with large numbers of specialized cores, ASIPs (Application-Specific Instruction-set Processors) and accelerators, each optimized for energy efficiency on specific tasks. However, this trend poses significant challenges for system design: conception and design of the accelerators, finding the right mixture of cores for current and future workloads, providing the right interconnections between different cores, managing power budgets when only a fraction of cores can be turned on at any given time, and validating that the wide variety of cores are correct.

Stakes

Heterogeneity is the most viable way forward to ensure continued growth in computing systems performance without a miraculous improvement in device-level energy efficiency. Failure to enable this path will severely limit our ability to leverage future device scaling to improve performance.

Actors

- **Industry:** Hardware developers and integrators need to determine the right mix of processors, accelerators, and interconnect, and need to define standards for interoperability at the software and data levels. CAD tools manufacturers should deliver tools helping further the developers of new accelerators.
- **Academia:** Explore the mix of processors, accelerators, and interconnect for future application domains and future technology nodes.

Technical Challenges

Finding the right "degree of specialization/flexibility" for specialization to be affordable. System-level integration of heterogeneous cores. Integration of heterogeneous IP.

HiPEAC Challenges

- **Efficiency:** Choosing the right mix of processors, accelerators, and interconnect. Efficient data movement support. Efficient system integration: SIP, 3D-stacking. Tools for design and validation of domain specific accelerators. Automated design space exploration to select the optimum hardware structures. Standardization of hardware and software interfaces. Reconfigurable cores.
- **Complexity:** Models for reducing/hiding heterogeneity. New approach to reduce simulation and validation time. Standard interfaces. Virtualization support for accelerators. Hardware support for software. Shared/coherent or virtual address spaces across cores and accelerators.
- **Dependability:** Redundant cores for reliability. Secure cores for security. Predictable cores and memory systems for safety-critical systems. Verification of interconnects and combined functionality.

Metrics

Maximum efficiency (operations/Watt) for key application. Ease of compilation. Scalability of interconnects.

Timeline

- **Short:** Development of efficient accelerators.
- **Medium:** Tools for improving productivity during development.
- **Long:** Automatic porting of legacy applications on parallel and heterogeneous systems.

Opportunities and Potential Disruptive Technologies

New forms of computing elements (PCMOS, 3D, Neuromorphic elements, etc.) and the integration of more traditional ones (FPGAs, GPU, CGRA, etc.) will be more common. New memory and interconnect technologies (photonic on silicon, 3D stacking, non-volatile memories, etc.) will alter the data/compute balance. Industry convergence on low-level programming systems (e.g., OpenCL) and virtualization for accelerators will increase adoption.

Potential Problems

Lack of programming models for heterogeneous systems. Difficulty of large-scale simulations. Lack of standard benchmarks.

4.1.2. Locality and communications management

Drive

As computing systems become increasingly complex the "distance" between processors, storage and data is increasing. This is not only a performance issue, as it takes time to move data, but more critically a power problem, as communications accounts for the majority of the total power in modern systems. However, experience has shown that while explicit data movement can give tremendous efficiency and performance benefits, the difficulty of manually managing data movement is prohibitively high for most developers. To effectively address these problems we must develop intelligent techniques for managing data placement and movement. Such techniques must be designed together with hardware resources to efficiently store and transport data. This will also impact the current thinking of "best effort", "as fast as possible" processing toward a more "on-time" model, "processing only when required". Ultimately, the memory hierarchy should be revisited to enable a co-location of computing and storage. New storage elements, if technically successful, will be a major player for this evolution.

Stakes

Ability to obtain high efficiency and performance from future systems. Ability to cost-effectively develop efficient software for large and complex systems.

Actors

- **Industry:** Hardware vendors must provide support for explicit data movement. Compiler manufacturers must expose this to the application and runtime, but not require it.
- **Academia:** Optimizations for data movement spanning embedded to HPC. Runtime systems and compilers for intelligent data placement and movement. New concepts, architectures, and devices to enable co-location of computation and storage.

Technical Challenges

Automatic design of the optimal memory hierarchy for heterogeneous computing systems. Design of simple but effective performance models that help hardware designers and programmers to make the right decisions. Static and runtime systems for automatic intelligent data movement. Revisit the "best effort" paradigm and the memory hierarchy scheme vs. explicit scheduling. Leverage advances in new storages devices. Develop new storage devices allowing co-location of processing and storage.

HiPEAC Challenges

- **Efficiency:** Optimizing data movement. Controlling hardware prefetchers/DMA engines. Intelligent runtime systems for data movement. Memory hierarchy design for both explicit and implicit data movement. Minimizing coherency overhead with explicit data movement. Cache management. New architectures for co-located computation and storage.
- **Complexity:** Modeling performance/energy costs of data movement. Tools to automate runtime and static data movement decisions. Tools for PGAS system data movement. Debugging and performance analysis support. Legacy code migration support. Mitigating NUMA effects and variable latency accesses.
- **Dependability:** Correctness guarantees for data movement with concurrency. Memory models for coherency and message passing. Handling device failures. Quality of service in virtualized/shared resource environments. Predictable latency for safety-critical systems. Impacts of shared memory resources on multi-core performance. Revisiting the "best-effort" paradigms towards a more "on-demand" processing.

Metrics

Percentage of data from explicitly-managed transactions. Speedup from explicit communications. Portability of explicitly-managed memory code. Power reduction from simplifying the memory hierarchy.

Timeline

- **Short:** Simple models for manual locality management.
- **Medium:** Automatic static locality management, "on-demand" processing approaches. Architectures with reduced memory hierarchy.
- **Long**: Automatic and dynamic locality management. Use of new storage devices for co-locating storage and processing.

Opportunities and Potential Disruptive Technologies

New memory and interconnect technologies (photonic interconnect, stacked die, etc.) will alter the optimal design point for memory systems. Non-volatile memories might eventually blur the line between primary and secondary storage. Higher-level domain-specific programming systems will enable easier runtime/static analysis.

Potential Problems

Complex access patterns remain difficult to optimize. Hardware programmability for explicitly managed communications. Lack of integrated hardware/compiler design research. Changing the mindset from "as fast as possible" into "only when necessary".

4.2. System complexity

4.2.1. Cost-effective software for heterogeneous multi-cores

Drive

The transition to ubiquitous heterogeneous parallel processing is the path forward to tackle computing power efficiency. However, this hardware solution comes at an enormous cost in program complexity: parallelizing applications, mapping computation to heterogeneous processors, and adapting to new systems and architectures. Today, the cost of these development activities is prohibitively high for virtually all developers, and it will only increase as systems become more parallel and more heterogeneous. To enable companies to leverage the potential of these future systems, we must develop tools to that manage this complexity for the programmer. Such tools must provide simplified interfaces for writing software, guarantees of performance portability across systems, and a path for integrating legacy code.

Stakes

Ability to cost-effectively leverage future performance growth in computing systems for new and existing applications.

Actors

- **Industry:** Software developers need tools to leverage new hardware. Hardware designers need tools to make new hardware usable. Compiler developers need to standardize interfaces and extensions to make code portable and debuggable. Everyone needs to address the issue of moving legacy code to new systems.
- **Academia:** New programming systems and approaches. Runtime and static optimization strategies for complex architectures. Scaling from embedded SoCs to HPC. Interoperability with legacy code and systems.

Technical Challenges

Performance portability. Running code on heterogeneous devices. Co-designed virtual machines. Data movement. Runtime/static optimization and load balancing. Programmer feedback. Debugging. Correctness. Legacy code on new systems. Programming models for specialized architectures.

HiPEAC Challenges

- **Efficiency:** Performance portability across different systems. Runtime/static optimization. Runtime performance monitoring and analysis. Profile guided JIT compilation and optimization for managed languages. Runtime timing analysis for latency requirements.
- **Complexity:** High-level software development with performance portability across different systems. Auto-analysis and parallelization of complex loop nests. Programmer feedback for understanding performance and power. Debugging support. Modeling and predicting power and performance. Design space exploration tools to help select the best architecture and compilation options. Shared resource modeling and analysis. Integrating with legacy code and workflows.
- **Dependability:** Formal correctness of runtime systems and user code in the presence of concurrency. Handling device failures. Quality of service in shared/virtualized environments. Programming models for ensuring timing for safety-critical systems on heterogeneous systems and accelerators.

Metrics

Percentage of peak performance/efficiency automatically achieved for a given application across multiple systems. Ease of obtaining high efficiency on different systems. Ability to integrate and accelerate legacy code.

Timeline

- **Short:** Systems to enable understanding of existing code and assist with parallelizing. Directive-based compiler tools. Runtime analysis.
- **Medium:** Programming systems with a more integrated runtime and language. Providing object-oriented paradigms across accelerators. Integrated performance/power/timing modeling and optimization.
- **Long term:** Full performance portability. Self-adapting software. New programming paradigms.

Opportunities and Potential Disruptive Technologies

Standard access layers to diverse devices (e.g., OpenCL) provide a good low-level platform for research and tools. The LLVM compiler tool chain provides a modern accessible base for new compiler development and research. Polyhedral loop transformation frameworks are becoming mature. Heterogeneous systems are becoming standard with CPU+GPU+video codec in nearly every device today. Domain specific languages have the potential to accelerate adoption. New programming paradigms or self-adapting software will hide the complexity to humans, but their final behavior should be under control (e.g. by meta-rules).

Potential Problems

Legacy code still dominates and is hard to understand. Hard to get realistic problems into academia. No representative heterogeneous benchmark suites. Performance prediction is becoming harder with new technology.

4.2.2. Cross-component/cross-layer optimization for design integration

Drive

The decoupling of design and production, combined with increased levels of on-chip integration, are leading to system-on-chip designs with increasing numbers of components from wider varieties of vendors. In addition to the hardware components, larger portions of the software stack are being provided as components from companies or open-source projects. This complex integration leads to significant inefficiencies across the component (block-to-block) and layer (hardware/software) boundaries. To produce efficient products with this approach, we must develop tools, standards, and methodologies that enable optimization across these boundaries. Such tools must be able to understand and manipulate the interaction of hardware components, software systems, and design constraints such as timing, power, and performance, across components from different vendors.

Stakes

Ability to cost-effectively design efficient products with multiple vendors' IP. Ability to optimize across complex systems, in particular with shared resources.

Actors

- **Industry:** EDA tool manufacturers and IP vendors need to standardize interfaces for optimization. Software vendors need to develop systems for optimizing across library boundaries.
- **Academia:** New runtime and static optimization strategies for complex architectures and software.

Technical Challenges

Components are provided as black boxes, but optimization must cross the boundaries. Black boxes obfuscate the high-level behavior that is often critical for efficient optimization. Specification of the non-functional properties of the components (e.g. temporal behavior, data pattern scheme, power profile). Multi-criteria optimization (e.g., latency plus energy). Multi-modality optimization (e.g., software plus hardware). Runtime software optimization, dynamic binary translation. Optimizing hypervisors.

HiPEAC Challenges

- **Efficiency:** Power and performance modeling of hardware and software at design time. Optimization taking into account runtime behavior. Electronic System Level (ESL) power modeling.
- **Complexity:** Understanding black-box IP components. Standard interfaces for optimization. Software development techniques that allow cross-library optimizations of components at runtime and at link time. Optimization of virtualization layers.
- **Dependability:** Correctness of optimizations. Multi-objective optimization respecting application-level constraints (e.g., timing, power, performance, QoS).

Metrics

How deep into a component the optimization can go. Accuracy of model-predicted behavior vs. post-place-and-route hardware simulation. Speed of optimization and impact on industrial workflow.

Timeline

- **Short:** Semi-automatic cross component/cross layer static optimization.
- **Medium:** Automatic cross component/cross layer static optimization.
- **Long:** Automatic cross component/cross layer static with dynamic runtime optimization.

Opportunities and Potential Disruptive Technologies

ARM's European presence and customer knowledge could be a large benefit if information can be shared with researchers in a non-restrictive manner. Multiple SoC designs on-going in Europe. Advances in convex optimization need to be more heavily leveraged by the computing systems community.

Potential Problems

Dramatic increase in ASIC cost reduces the number of customers for such tools. Post-place-and-route power/performance analysis is essential for accurate evaluation, but is very difficult and expensive for academic teams to accomplish. Virtualization layers will be difficult to analyze and optimize.

4.2.3. Next-generation processing cores

Drive

Processing cores form the heart of all computing systems. Efficiency constraints are forcing us to design systems with large numbers of task-specific (heterogeneous) cores. This future requires three key trends for next-generation processing cores: lower power, lower verification cost, and more intelligent reliability. For overall system efficiency, we must design efficient cores. This trend makes the complex structures of the past less attractive, and encourages a move towards simpler designs. As systems will be built of a variety of task-specific cores, we need to reduce the per-core design and verification costs. And since there will be hundreds or thousands of cores per chip, the ability to ensure reliability in the face of manufacturing variability and unreliable components becomes critical.

Stakes

Ability to produce the energy-efficient systems needed to leverage the increasing numbers of available transistors. Ability to provide predictable behavior for safety-critical systems. Ability to provide reliability in the face of ever increasing variability and failure rates in newer technologies.

Actors

- **Industry:** Chip and IP block designers (ARM, STMicroelectronics, ST-Ericsson, etc.) need to push efficiency and reliability while minimizing development cost. EDA tool vendors need early and accurate power/performance modeling and higher-level functional verification and design methodology.
- **Academia:** New efficient architectures. Verification techniques for hardware. Power/performance modeling.

Technical Challenges

Minimizing the cost of data movement within chips (register-register/cache-register/memory-cache). Optimizing computational resources for applications. Determining how much to specialize. Handling process variation. Handling hard/soft errors at smaller feature sizes. Better energy managament than using DVFS (Dynamic Voltage Frequency Scaling). Providing usable architectures for compilers. Common generation of hardware and its programming stack. Higher level hardware design tools (e.g. C++ based).

HiPEAC Challenges

- **Efficiency:** Minimizing data movement within cores (register-functional unit, register-register, register-cache). Co-locating processing and storage. Optimizing computation unit design and selection. Optimizing data path widths. Accuracy/power tradeoffs at the architectural level. Custom and reconfigurable data paths/functional units.
- **Complexity:** Improving hardware verifiability. Enabling programmability through compiler-targetable designs. Enabling virtualization for accelerator cores. Advanced hardware performance monitoring. Enabling concurrent debugging. Supporting legacy applications and binaries. Higher level hardware design tools (e.g. C++ based).
- **Dependability:** Process variability and unreliable components. Providing predictability for time-critical applications. Providing security for secure applications. Correct by construction design methodology. Formal proof of correct behavior for hardware and software.

Metrics

performance per Joule; performance per byte from main memory; average percent of maximum performance obtained automatically by compilers; quality degradation under process variability.

Timeline

- **Short:** Energy efficient movement within cores, tools to assist in the generation of the hardware of the computing cores and their compilers.
- **Medium:** Automated design space exploration tools to propose efficient architectures and compilers.
- **Long:** New compute engines with minimized data movement.

Opportunities and Potential Disruptive Technologies

New forms of computing elements (PCMOS, 3D, bio-inspired computing elements, etc.) and the integration of more traditional ones (FPGAs, GPU, etc.). New memory and interconnect technologies (optical, stacked die, non-volatile, etc.) will alter the data/compute balance. New application demands will shift the focus of the cores.

Potential Problems

Core power may be small compared to the surrounding infrastructure and data movement. Current processors are becoming more complex and harder to understand and to work with. Very few teams can design processors through place-and-route to get credible performance results. Adoption of new processors is very slow. Testing processor designs requires full system and compiler infrastructure.

4.3. Applications

4.3.1. Architectures for the Data Deluge

Drive

The world creates, stores, and processes a staggering (and increasing) amount of data today. In addition, the complexity of the data is increasing, as is the sophistication of the required processing. Yet buried within this data are key insights into business, society, health, and science. To transform this

deluge of data into value requires computing infrastructures that can process it in real time. Today's systems struggle to keep up, and projected increases in data far outstrip projected growth in processing power and storage. Addressing this divergence requires developing systems and techniques that enable us to store and process data with orders of magnitude more efficiency and methodologies to program them to ensure real time response.

Stakes

Ability to extract value from the massive streams of digital data in today's society. Ability to handle the ever-increasing data volumes of the future.

Actors

- **Industry:** Data centers need to improve the scalability, capacity, and performance of their processing and storage systems.
- **Academia:** Develop new algorithms to efficiently extract information from the data streams. Real-time, best-effort processing methodologies with statistical – or formal – guarantees of correctness and latency.

Technical Challenges

Volume of data (storage, retrieval, transportation). Processing requirements (throughput, performance). Latency requirements (guaranteeing latency, best-effort calculations, ensuring uniform latency). Scalability (power, latency).

HiPEAC Challenges

- **Efficiency:** Energy efficient processing of streaming data sets. Cost of moving data. Processing in-place. Choosing the right location for processing based on current constraints (battery, communications cost). Finding new computing paradigms better suited to the natural data processing (Recognition, Data-Mining and Synthesis).
- **Complexity:** Large-scale data/processing orchestration. Latency control.
- **Dependability:** Enabling aggregate analysis while maintaining privacy. Reliability in the face of best-effort calculations. Enabling commodity use of cloud services to provide vendor diversity.

Metrics

Volume of data processed per Joule. False positive and false negative rates for advanced recognition algorithms. Energy/quality tradeoff for best-effort calculations.

Timeline

- **Short:** Energy efficient architectures for data processing. Latency analysis tools.
- **Medium:** System development tools for minimizing latency and energy. New concepts and processing paradigms for natural data processing.
- **Long:** Real-time analysis of data. Accelerators for natural data processing using new computing paradigms.

Opportunities and Potential Disruptive Technologies

New memory and interconnect technologies (silicon photonics, stacked die, non-volatile memories, etc.) will alter the data/compute balance. Computation embedded in the storage system and non-volatile storage embedded in the processors. Interoperability between cloud providers may spur innovation on the backend to differentiate. New computing paradigms that are efficient for non-exact data processing, such as bio-inspired, stochastic, probabilistic will require different architectures and programming models for efficient implementation.

Potential Problems

Large-scale data applications are not open to academics. Data is often proprietary. Evaluating real-time behavior at scale requires complex testing setups and infrastructure. Inertia to move away from classical processing approaches.

4.3.2. Reliable systems for Ubiquitous Computing

Drive

As computing systems become smaller, more powerful, and universally networked, they permeate even deeper into all aspects of society. These systems are now essential for safety, efficiency, and social interaction, and must meet demands for higher levels of reliability. This encompasses everything from correctness and dependability of safety-critical systems to availability of social networking services and power distribution networks, and privacy and security for personal and corporate data.

Stakes

Ability to provide reliable systems in the presence of unreliable technology. Ability to continue growth in the mobile sector. Ability to ensure reliability and privacy for mobile applications and infrastructure. Ability to ensure safety for embedded infrastructures.

Actors

- **Industry:** Needs to develop ubiquitous computing standards. Needs to improve the reliability and the security of the software and the hardware components.
- **Academia:** Needs to work on techniques to automatically verify and design security and safety properties of whole systems. Need to ensure the transfer of these techniques to real-world systems and problems.

Technical Challenges

Complexity of the systems. Perseverance of hackers. Increasing reliability problems with smaller feature sizes. Coping with the dispersion of characteristics of basic components. Moving away from the "worst case design" methodology.

HiPEAC Challenges

- **Efficiency:** Load balancing over the complete system. Energy scavenging for sensor networks. Extreme low power for implanted systems. New approaches at architecture level to dynamically detect errors of components. Moving away from the "worst case design" methodology allowing more efficient designs while ensuring predictability.
- **Complexity:** Large scale distributed system. Correctness (timing, testability, composability) guarantees. Interoperability. Ensuring quality of service across integrated components.
- **Dependability:** Graceful degradation in the presence of failing components. Security and safety guarantees. Isolation of software domains.

Metrics

Number of security fixes, hacks. Tolerating device variability. Tolerating device faults. Achieved utilization under safety-critical constraints.

Timeline

- **Short:** Manually secured and verified systems.
- **Medium:** Semi-automatically secured and verified systems. First designs with variability and fault tolerance.
- **Long:** Fully automatically secured and verified systems, or correct-by-design tool chains. Self-reconfiguring systems to optimized variability and errors.

Opportunities and Potential Disruptive Technologies

Quantum computing for security applications. Leveraging parallelism to improve deterministic execution.

Potential Problems

Fundamental security mechanisms broken (e.g. crypto made worthless by quantum computing). Difficulty of achieving predictability on commodity processors with shared resources. Need to work at higher levels of abstraction for efficiency while still ensuring low-level reliability. Gap between theoretical work on timing properties and industrial practice.

5. Conclusion

Several aspects of the future of computing systems for the next several years are clear:

- **Energy efficiency** will force hardware to move to heterogeneous parallel systems
- The **Data Deluge** will drive applications
- Ubiquitous computing will force a business towards **clouds and mobile devices**

Yet these same trends lead to significant challenges:

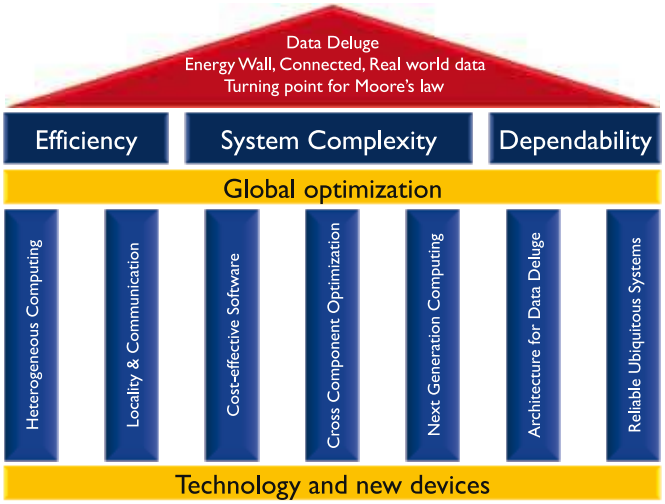
- Heterogeneous systems are prohibitively difficult to program
- Existing infrastructures for data processing will not scale up
- The focus on mobile and cloud will result in significant reliability challenges

Based on these trends and challenges, HiPEAC has identified three Core Computing Challenges:

- ***Efficiency:*** Maximizing the computation per unit of energy
- ***Complexity:*** Providing tools to enable software development for new systems
- ***Dependability:*** Ensuring reliability and predictability for ubiquitous computing.

Each of these challenges plays an integral role for the future growth of our computing capabilities and the societal benefits we derive from them. To address these challenges, HiPEAC has identified three key areas for research and seven specific research objectives:

- **Efficiency** (with a focus on energy efficiency)
  - Heterogeneous computing systems
  - Locality and communications management
- **System Complexity**
  - Cost-effective software for heterogeneous multi-cores
  - Cross-component/cross-layer optimization for design integration
  - Next-generation processing cores
- **Dependability and applications** (with a focus on their non-functional requirements)
  - Architectures for the Data Deluge
  - Reliable systems for Ubiquitous Computing



More and more, it will become critical as well to investigate research directions breaking with the line of classical Von Neumann systems and the hardware/software boundary to adress these challenges.

By focusing on these areas, the HiPEAC community will be able to make significant high-impact contributions to computing in Europe. These advances are necessary to enable our society in the 21st century to continue to reap the benefits of computing systems that have so revolutionized the 20th century.

The HiPEAC 2011/2012 Roadmap In-depth

A. Key Trends and Challenges in Computing Systems

To analyze the trends and challenges facing computing systems in the beginning of the 21st century we have considered four key stakeholders: **society, business, applications,** and **systems technology.**

A.1. Societal Trends and Challenges for Computing Systems

Computing Systems R&D helps address Europe's key socio-economic challenges, from a lower carbon economy, to health and well-being in an ageing society, competitive businesses and manufacturing for a sustainable recovery, and learning and sharing of cultural resources [ICTWORK]. For decades to come, we consider the following nine essential societal grand challenges [ISTAG], which have deep implications for computing, and vice versa.

Energy: computing systems are both part of the growing energy problem (consuming about as much energy as civil aviation) and our single most effective tool towards its solution. To improve the energy consumption of computing systems we must improve its efficiency. At the same time, the use of computing systems to model, analyze and optimize our existing and future energy production and consumption infrastructures and technologies will have an even bigger impact. To enable new advances in energy efficiency and production we must continue to improve our computational capabilities.

Computing Systems Challenges: improve energy efficiency to reduce Computing's energy footprint; increase computational capabilities to enable better tools for modeling and design.

Transportation and mobility: Modern society depends on inexpensive, safe and fast modes of transportation. However, transportation is an environmental hazard, average speeds are low, and tens of thousands die every year in transportation accidents. Computing is a key enabler for improving mobility by providing the technology to optimize and control traffic flows, monitor and optimize fuel usage, and provide advanced active safety features. Besides improving transportation, computing systems also help us avoid it by providing virtual interaction through email, instant messaging, and video conferencing, all of which reduce the need for physical travel.

Computing Systems Challenges: provide efficient computation to enable sophisticated processing and control; ensure ubiquitous communication to enable large-scale optimization; guarantee dependability for safety-critical operation.

Healthcare: The use of computing systems technology is essential to improve healthcare. There is a great need for devices that monitor health, assist healing processes, and identify early-stage diseases. These devices can both improve the quality of care and reduce cost. Further, as more health

Carbon Footprint of Computing Systems

According to Gartner [Gartner 2007], computing systems were estimated to account for approximately 2% of global carbon dioxide (CO_2) emissions, roughly the same as aviation in 2007. This corresponds, for example, to 60g of CO_2 per hour a desktop computer is turned on, or 0.0003 kWh of energy or 0.2 grams of carbon dioxide for every Google search that is run [Google2009].

Making computing systems itself more energy-efficient will therefore significantly contribute to the energy challenge, and this is a theme that runs throughout our vision. However, despite its high-energy consumption, computing is already an enabler for energy reduction in other domains. Data centers, in particular, actually represent a net saving (up to 4x their CO_2 emissions) as people use online, rather than physical, services [Wehner2008]. This includes online media that reduces the volume of paper being used and post being sent; e-commerce, that results in fewer physical journeys to shops; video conferencing and teleworking, which reduce business trips. Computing systems also drive optimization and design for better energy efficiency, e.g. for more efficient cars designs, engine control, and energy generation optimization. It is the core of efficient control of energy in all aspects.

information becomes available online, the analysis of medical history data to identify underlying causes and societal threats, such as pandemics, is becoming an increasingly valuable tool.

Computing Systems Challenges: provide efficient computation for low-power devices and for large-scale research; enable ubiquitous communication to provide for the integration of patient and societal data; guarantee dependability for health-critical systems and security for private information.

Aging population: Life expectancy has increased considerably over the last century and continues to do so even today. As a result, the need for healthcare and independent living support is growing significantly. These services allow people to remain independent by providing assistance through technologies such as household robots and advanced home automation. By providing the infrastructure and processing capabilities to enable these assistance technologies, computing systems will reduce the cost for caring for an aging population and improve their quality of life.

Computing Systems Challenges: provide efficient computation for mobile home devices; enable ubiquitous communication for remote monitoring; guarantee dependability for health-critical systems.

Environment: computing systems play a critical role in protecting the environment by modeling, controlling, and optimizing our impact. Automobile engines, agricultural pesticide use, power generation, and traffic flow all require advanced systems to continuously monitor the environments to maximize efficiency and minimize pollution. In addition to providing the infrastructure for enabling these optimizations, computing systems provide scientific resources for modeling the effects of environmental change and minimizing it, through better design across all aspects of society from buildings to transportation to plastics and water treatment.

Computing Systems Challenges: provide efficient computation for small sensing applications and large-scale simulation and modeling; enable ubiquitous communication for collecting data across large networks of sensors.

Productivity: In order to remain competitive, economies have to continuously improve the productivity of their industrial and non-industrial processes. Computing systems have contributed dramatically to productivity increases across all industries by helping to automate design, assembly, inspection, packaging, and forecasting. To achieve greater productivity gains these existing systems must become more sophisticated and cheaper. The continuous cost and performance pressure on these systems can only be met by dramatic improvements in computational efficiency and system robustness.

Computing Systems Challenges: provide efficient computation for both embedded and large scale systems; enable ubiquitous communication for collecting accurate product usage and maintenance data; guarantee dependability of manufacturing systems under tight resource constraints and reliability of end products.

Safety: Safety-critical systems today are controlled by computing systems. Building dependable systems requires handling failing components, addressing complex timing constraints, and ensuring functional correctness at design time. For safety-critical systems, availability, integrity and maintainability are essential. Yet for competitiveness these systems must increase functionality and reduce cost. Such pressures make the verification of timing and functional constraints both increasingly difficult and increasingly important. These requirements can

only be met by developing more powerful and expressive techniques for analyzing and understanding real-world systems and problems.

Computing Systems Challenges: provide efficient computation for cost-constrained embedded systems; guarantee dependability and correctness across all layers of complex systems.

Security: Advances in computing are a double-edged sword for security. For law enforcement and national defense modern computing systems provide sophisticated means for analysis, detection, and forensic investigation. Yet these same advances are putting increasing amounts of personal data in the open and making it harder for individuals to control how this data is used. With computing systems proliferating in all aspects of life, dependable and secure computing must be a cornerstone of modern infrastructure.

Computing Systems Challenges: provide efficient computation for portable law enforcement and defense systems as well as large-scale data analysis; guarantee security and traceability of sensitive and private data.

Education: Higher education is the foundation of our information society. Technical profiles typically require 3-6 years of higher education, for fields where the core technologies change every 2-8 years. As a result, the focus of computing education should be to prepare the graduates for life-long learning, and to use computing systems to make the learning process more effective and more efficient (interactive, student-centered). Even for students who do not focus on technical profiles, Europe must ensure that it educates them with the literacy required to leverage advances in computing systems to improve productivity in their field. Therefore, computing systems must be both taught as a subject unto itself, and as the core tool which all other subjects leverage. Such computer literacy is one of the key ingredients for the economic success of Europe in the future.

Computing Systems Challenges: provide high-level access to computational tools for non-ICT experts; provide exposure to cutting-edge technology to technical students.

A.2. Business Trends and Challenges for Computing Systems

Information and Communications Technology has been one of the major drivers for productivity improvements and new opportunities in business over the past decades. The number of new technologies and services being brought forth today is staggering. And behind it all, the market structure for production and development of new technology is undergoing a major shift.

The desktop PC is no longer driving the computing industry. The computer industry is moving into a commodity market from a high-tech market, and it has drastic impact on the market composition. Past leaders in the PC market, like HP, IBM, Compaq, Gateway are giving up, even if they were market leader (like HP). The PC market players can only compete on cost, as hardware and software are no longer differentiating factors. For consumers, new form factor like tablets and smart phones are cannibalizing the market. According to Gartner, the Western Europe PC Market Declined 19 Percent in Second Quarter of 2011. One exception to this trend is Apple, the first worldwide company by its capitalization, which controls its entire value chain and has its own ecosystem.

Computing Systems (Anti-) Challenges: Only one or two companies will develop processors for the PC market, and they will carry out most of their own research.

Energy efficiency and mobility define the market: Laptops are better selling than desktops for most consumers, and smart phones and tablet are attracting more and more consumer than PCs. Even for games, the PC is becoming a less important market, replaced by dedicated gaming devices and mobile phones. For most consumers, PC use is now limited to running an of-

The End of the PC Era

"There's a clear secular movement in the consumer PC space. The impact of the economy has impacted consumer sales."
"For our PC business to remain the world's largest personal computing business, it needs the flexibility and agility to make decisions best for its user direction."
"The tablet effect is real; our device has not gained enough traction in the marketplace with consumers and we see too long of a ramp-up in the market share."

"Due to market dynamics, significant competition, and a rapidly changing environment – and this week's news only reiterates the speed and nature of this change – continuing to execute our current device approach in this marketplace is no longer in the best interest of HP and HP shareholders."

HP's CEO Leo Apotheker, announcing HP's failure to enter the tablet market. (August 18th, 2011)

Western Europe: PC Vendor Unit Shipment Estimates for 2Q11 (Thousands of Units)

Vendor	2Q11 Shipments	2Q11 Market Share (%)	2Q10 Shipments	2Q10 Market Share (%)	2Q11-2Q10 Growth (%)
HP	3,171	25.1	3,376	21.6	-6.1
Acer Group	2,046	16.2	3,696	23.7	-44.6
Dell	1,371	10.8	1,571	10.1	-12.7
Asus	1,021	8.1	1,324	8.5	-22.9
Apple	879	7.0	875	5.6	0.5
Others	4161	32.8	4751	30.5	-12.4
Total	**12,649**	**100**	**15,593**	**100**	**-18.9**

Note: Data includes desk-based PCs and mobile PCs. Media tablets are excluded. Source: Gartner (August 2011)
(from http://www.gartner.com/it/page.jsp?id=1769215)

fice suite. Most of its other functions, including communication, sharing images, videos and chatting, are being replaced by mobile devices that can be used anywhere, anytime. The use of "cloud" resources will push this trend even further. As a result, the market will see most growth in the cloud and mobile devices, which will push processor design resources in those directions. In both areas power efficiency is the dominant focus.

Computing Systems Challenges: Power efficiency will be the key performance metric of future processing devices.

Platform convergence: There is a dramatic move towards platform convergence, particularly in the exploding smartphone/tablet space. While these small form factor computers ran a wide variety of operating systems in the past, today's market is dominated by two players (iOS and Android) who are pushing complete platforms (operating system and processor architecture) across multiple devices and form factors.
In the cloud space, providers are actively working to standardize infrastructure systems to enable interoperability.

In the architecture domain, the historic divide between x86 architectures for desktop and high-end computing and ARM/DSP for embedded low-power devices is shifting. ARM is looking to leverage its power efficiency advantage in the high-end market while Intel is looking to push its performance advantage in the embedded space. At the same time all manufacturers are including programmable graphics processors and custom accelerators to improve energy efficiency. This move is resulting in a convergence on heterogeneous architectures across all facets of computing systems.

This commoditization of functionality impacts the market: customers are no longer loyal to a particular brand as long as the product's platform (be it Web 2.0 or mobile phone OS) can do the job. They easily switch brands, because the platform ensures that you do not lose your investment in data. Since people buy new equipment/software every 3-4 years, the market becomes very volatile. There is no guarantee that a consumer's next system will be from the same brand. A big player can turn into a niche player in just a couple of years. The recent story of Nokia's downfall in the face of Apple's rise to dominance is a telling example. Companies have to be extremely innovative to differentiate themselves in an environment where everyone is selling the same platform.

Computing Systems Challenges: use the convergence, don't fight it. The challenges are to build most energy and application efficient systems around the dominant platforms (hardware and operating system).

Global or specialized companies: The semiconductor industry is slowly changing from a high-tech into a commodity industry: chips and circuits are everywhere and need to be low cost. Further, as devices shrink the cost for producing and verifying chips is becoming so high that only the highest volume devices can be profitable. These trends are leading to de-verticalization and specialization. Instead of having companies controlling the complete product value chain from hardware design through software and packaging, the trend is to split big conglomerates into smaller companies. Each of these companies can then specialize in their competence domain, thereby producing cheaper products in a more agile manner. For example, many large companies have spun off their semiconductor divisions, and in turn the semiconductor divisions spin off their IP creation, integration and production, thus becoming "fabless" or "fablight". Examples are Siemens, Philips, and, in the past, Thomson.

This specialization to smaller companies allows significant consolidation within competence domains through merger and

Power Constrained Computing

Fitting more computing capacity into a limited power envelope is one of the key challenges facing data centre designers today. It impacts companies whether they are building a £500 million data centre or simply working how much compute that can fit into a couple of racks at a shared facility.

One of the emerging approaches to solving this problem is to look at the technologies in low power-consumption appliances like phones and applying them to dense clusters in server-like configurations. Whether it is in smartphones, tablets or other embedded systems, the processor at the heart of these low power devices is generally ARM-based.

With Ubuntu Server becoming the de-facto standard for cloud infrastructure and big data solutions, we recognize that power consumption is key to efficient scaling. Building on four years of working with ARM, we are now taking the step of supporting Ubuntu Server on ARM. We expect these processors to be used in a variety of use cases including microservers.

August 16th, 2011 by Chris Kenyon - vice president of OEM services at Canonical.

(from http://blog.canonical.com/2011/08/16/armserver/)

acquisition. Through such consolidation companies can obtain a critical mass in their competence area, which can be essential for profitability in low-margin commodity industries.

Importantly, a commoditized, horizontal market requires product integration between multiple companies for each product value chain. A modern embedded device may include hardware, IP, and software from many different vendors, all integrated into the same device. For these parts to work together, standardization of tools and interfaces must be accomplished.

However, commoditization has the downside that it is difficult to do system-wide optimizations. As each component is delivered as a black box, the ability to optimize across component borders is limited. The benefits of such full system optimization can be seen in the success of two exceptional companies that intentionally, and aggressively, own the full value chains from hardware through software to retail: Apple and Google.

In the emerging tablet business, Apple is a special player: it has its own hardware, operating system, application shop and ecosystem of developers. Despite being historically considered a premium brand, this vertical integration allows it to produce products at prices that even low-cost companies have trouble beating. HP gave up in the tablet market after only a few months. Google has recently recognized the benefits of such vertical integration and acquired Motorola's mobile phone division in an possible attempt to achieve similar benefits from integration.

The mobile phone market of the past three years poses a particularly interesting test of the benefits of commoditization. Google's Android platform has made it increasingly difficult for

The Exploding Design Cost

- Total SoC design costs increased 39% from the 32nm node to the 28nm node and are expected to increase 29% again at the 22nm node.
- Total SoC silicon design costs increased 35.7% at the 28nm node.
- Total Software design costs increased 42.5% at the 28nm node and are forecast to show a CAGR (Compound annual growth rate) of 138.9% through the 14nm node.
- Advanced Performance Multi-core SoCs represent the most expensive silicon designs with Multi-core SoCs and Basic SoCs exhibiting lower design costs.
- Derivative SoC silicon designs allow designers to accomplish their solutions at a fraction of the cost compared to first time efforts at the same process node when it first becomes commercially available.
- Costs for an Advanced Performance Multi-core SoC design, continuously done at the 45nm node, will experience a negative CAGR of 12.5% by the time the 14nm process geometry becomes commercially available, showing that subsequent designs at the same node become less expensive over time.
- **28nm silicon with a $20 average selling price is required to ship 6.521M units to reach the breakeven point.**

(from http://www.semico.com/press/press.asp?id=299)

Dying Arts and Fading Fabs

"The skills you need to close out the timing at the transistor level are becoming a dying art"

…

"As we go forward, and start worrying about very exotic processes that we're going to have to deal with in the future, those transistor skills are going to need to become very, very important once again. And as a designer, you're going to have to worry about everything – from architecture down to transistors."

Simon Seagers, Head of ARM's Physical IP Division, at Hot Chips 2011.
(from: http://www.theregister.co.uk/2011/08/20/microprocessors_may_face_trouble_ahead/page2.html)

mobile phone vendors to differentiate their products. All Android phones run the same programs and the same operating system. They all use Google's cloud services. This leaves Android manufacturers to compete on cost and has driven their profits very low, while the overall Android market share increases. At the same time, Apple's combination of unique hardware and software provides a significant differentiator, and their vertical integration allows them to obtain significantly higher margins on their products. As a result, they command a higher selling price, have higher margins, and make, despite being only one vendor, the majority of the profits in the market.

Computing Systems Challenges: develop interfaces and integration techniques that allow multiple vendors' products to be smoothly assembled; address the rising cost of modern chip design to enable lower-volume products to come to market and academic projects to be realistically tested; enable cross-border optimization when integrating black box components.

The economy of collaboration: The Internet has enabled new communities and collaborations to come together from across the globe. Individuals and companies are contributing their time and resources to sharing knowledge and expertise with others like never before in history. This phenomenon is increasingly visible in all computing domains. The examples with the highest impact are undoubtedly open source software products such as the Linux kernel and the GCC compiler tool chain, both of which are freely available, and both of which are primarily developed by paid employees of large corporations.

Linux and GCC have become the de facto standards in the computing industry due to their zero cost and rapid evolution into robust tools. This evolution came about due to the novel licensing terms under which they were released. The GNU General Public License (GPL) requires that whenever a modified binary program is distributed, the corresponding source code changes also be released under the GPL terms. This ensures that most enhancements from individual companies make their way back to everyone else. These licenses focus on the protection of the freedom to use, modify and redistribute content rather than on limiting their exploitation rights. Similar to applications distributed under traditional licenses, care must

The End of Free Scaling

Over the past 50 years, the continued miniaturization of transistors has allowed them to both run faster and use less energy. This meant that each new generation provided faster and more efficient computation. The entire computing systems industry, from hardware to software, relied on this "free scaling" to obtain incredible growth. Software and systems ran faster each year by simply swapping in newer hardware. But today devices are running into the hard constraints of the physical materials from which they are built: increasing speed increases energy consumption exponentially to the point where we are unable to cool the processors. This has put an end to the "free scaling" and we no longer see significant growth in speed or reductions in energy consumption.
The effect of the end of "free scaling" has been that manufacturers have started increasing the number of processor cores on each chip while reducing their speed. The motivation is that by reducing speed they can improve energy efficiency, and by increasing the core count they can improve performance. However, to take advantage of this increased performance, applications must be re-designed to leverage multiple processor cores. Such conversions are extremely expensive, and often impractical for large systems. But even if the computing systems industry can adapt to execute in parallel, continued device scaling is quickly leading us to the point where turning on the whole chip at once will not be possible anymore due to power and cooling limitations. This will force the systems to intelligently choose which parts of the chip to enable to obtain the best performance. To orchestrate this whole system effectively requires globally rethinking how computing systems are designed and built.

however still be taken that the license is compatible with the intended business uses.

Provisions in the GPL, which give recipients of derivative works full access rights to the incorporated source code modifications, can make it difficult for companies to leverage such products and maintain a competitive advantage. The ready availability of such software can also cause legal headaches for companies, forcing them to periodically audit their own software for uses of GPL code. Other free software and open source licenses with less demanding release terms, such as the Berkeley Software Distribution (BSD) license, also exist.

Computing Systems Challenges: leverage and extend open source software projects to reduce duplicated work, particularly within academia; develop competitive open source projects in the CAD space where none exist today; educate students on how to leverage open source material in commercial and academic settings.

Less is more: Consumers no longer demand only more features and better performance, but are increasingly interested in devices with acceptable performance at lower prices, in novel form factors, and with better battery life. This is particularly visible in the markets for small form factor devices such as smartphones, tablets, and ultra-small laptops, although high-density compute servers are feeling similar pressure. For mobile devices, consumers are now eager to trade off raw speed, configurability, and expandability for integrated experiences that deliver enough performance and improved ergonomics, battery life, and/or cost. These new pressures invalidate the performance-at-all-costs approach of the computing industry for the past several decades, and encourage a more efficiency-centric design and exploration of new technologies.

Not only does this trend push mainstream computing hardware towards smaller and more power-efficient designs, but it also impacts software. Now developers must write more efficient software for these less powerful devices. In particular, this requires leveraging the heterogeneous accelerators in these devices, such as video decoders and graphics processors, rather than relying on a fast CPU.

Today's most popular mobile platforms (iOS and Android in a certain way) further limit their users' ability to access the underlying system, and effectively require that applications be installed through a centralized application store, provided by the platform vendor. This approach trades off flexibility and freedom for security, convenience, and uniformity, and has proven to be a wildly successful approach. The clear lesson from this trend is that computing technology has reached a level of performance which is acceptable to most users and a level of complexity which must be largely hidden. Further, consumers are far more interested in the whole experience than the technical specifications.

Computing Systems Challenges: transition from the performance-at-all-cost mentality to functionality-at-lowest-cost; provide efficient processing architectures and software tools for leveraging them; provide security for mobile applications and data.

A.3. Application and System Trends Challenges for Computing Systems

The most direct way to see current trends in computing systems is to examine real applications and usage scenarios. Today, we live in a digital universe. We are witnessing a second revolution in information technology, where individuals, businesses, governments and societies alike rely on cost-effective, robust, and ubiquitous storage, communication and processing of unprecedented volumes of data — called the "Data Deluge" — for our daily needs. At the same time, the demand for intelligently processing that data is growing faster than technology advancements can sustain. To further compound the difficulties, existing computing systems technologies have hit diminishing returns in energy scalability, and energy has now become the defining performance characteristic.

We have identified four paradigm shifts facing computing systems applications over the next five years:

Data Deluge meets the Energy Wall in a Connected World

The combination of massive amounts of data, a demand for intelligent processing and ubiquitous communication, and increasingly constrained energy budgets lead us to summarize the trend in computing systems applications as: "**Data Deluge meets the Energy Wall in a Connected World.**" To meet the challenges posed by these trends we need to enable storage, communications, and processing with orders of magnitude less energy than we can today.

The Era of Connected Intelligence

"Semiconductor companies now need to become much more systems-oriented," Freescale CEO Rich Beyer said during Freescale's technology forum here this week. "We are into an era of connected intelligence where data is ubiquitous, and these devices [such as tablets] will conform to us and not have us conform to them, as in the case of the personal computer."

(from http://www.eetimes.com/electronics-news/4219247/Freescale-eyes-post-PC-era)

1. An unprecedented increase in the volume of data to process (data deluge),
2. A demand for far more intelligent processing of less structured data,
3. An absolute focus on power efficiency, either to reduce time between recharge for mobile consumer devices, or to make more compact and cost efficient servers for service providers
4. Ubiquitous connectivity across all electronic devices, both physically and between applications that share data.

These four paradigm shifts will force us to revisit all aspects of our present computing infrastructure to overcome the challenges in performance, scalability, security, and reliability that accompany this change.

Data Deluge: All facets of society are generating increasing amounts of data. Commercial sources include financial transactions, search histories, and product information. Public agencies contribute medical records, population databases, and legal and legislative data. Science and engineering routinely produce large-scale simulations for weather prediction, drug discovery, product development, and raw data from advanced sensors and experiments. And individuals are creating an unprecedented amount of personal data, greatly encouraged by the explosion of social networking. The unprecedented growth of data is forcing us to reevaluate how we work with data in computer systems.

This data is not only a byproduct of our digital society but also a valuable resource. Buried within this data are key insights into human behavior, market trends, disease, engineering safety, environmental change, and the basic workings of physics. Yet with such a massive deluge of data it is becoming increasingly difficult to get the most from it, and to do so in a timely manner.

To use this data we need to be able to quickly analyze and respond to it in real time. Rapid analysis is key not only for financial trading (where a millisecond advantage can allow extracting millions of Euros from the market before anyone else) and safety systems (where unknown delays can cause disaster), but also for delivering compelling customer experiences (by rapidly identifying trends) and for society as a whole (identifying pandemics and optimizing transportation). The challenge for the next decade will be in coping with this nearly infinite increase in data and the simultaneous demand for processing it faster.

Finally, as we populate the physical world with devices and sensors that will be connected to the cloud, they will start generating a constant stream of rich media, unstructured data, such as audio, video, and physical measurements of the world, as well as security and reliability data to detect threats and faults. All of this represents the new frontier of data deluge that our computing systems will have to deal with in the years to come.

Yet at the same time as we seek to make this data more available and more understandable, we need to provide solutions to protect privacy and confidentiality, to ensure the integrity of the data, and to properly authenticate users. These requirements must be in place to enable society to benefit from the collection of individual data without exposing the privacy of the individuals. Past experience has shown such requirements must be built-in as part of the basic system design and cannot be bolted on as an afterthought.

Computing Systems Challenges: develop storage systems with orders of magnitude higher power efficiency; develop low latency IO systems and intelligent memory hierarchies. Improve wireless and wired interconnect technologies.

Intelligent processing: turning data into information: An insatiable worldwide need for real-time delivery of information exists today. In this context, the challenge is to extract the required insights from massive datasets ("big") in near real-time ("fast") across a diverse set of structured, semi-structured and unstructured data sources ("total"). We refer to this as "*Big-Fast-Total*" data.
Such large data sets demand structured interpretation and visualization. The keyword searches of the past are no longer sufficient as the scale of the data is too large and the insights being sought are too complex. It is clear that today's organization need deeper insights across the board: 30% of business decisions are made on incomplete or not trusted information, and 50% of business leaders admit to not having access to the information they would want to have when they need to make decisions.

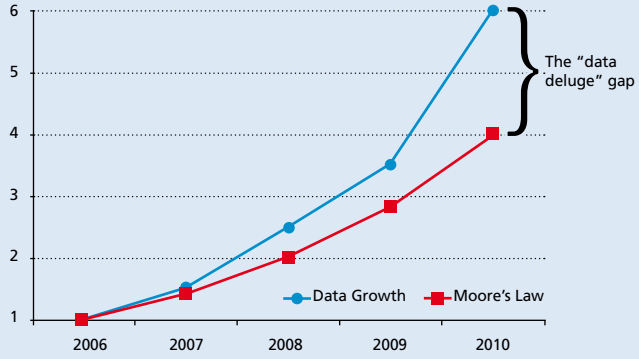
Data Deluge: a few Figures

The term "Data Deluge" was coined in 2003 [HT03] in the context of scientific data management to describe the massive growth in the data volume generated in research (and by scientific instruments), which was rapidly dwarfing all the data previously collected in the history of research. Since then, the ability to generate vast quantities of data has outpaced the infrastructure and tools to be able to support it in several other fields of society beyond the scientific community, including digital media (audio and video), commercial transactions, social networks, legal and medical records, digital libraries, and so on. The need to understand, organize, and sustain "big data" is today one of the highest priorities in information technology across disciplines, organizations and geographies [E10].

In 2010 the world generated over 1.2 Zettabytes (10^{21}) of new data, 50% more than it had in all of human history before that. To put this in perspective, 120 Terabytes of new data was generated in the time it took to read the previous sentence. In 2006, this amount was 200 Exabytes, 300 Exabytes in 2007, 500 Exabytes in 2008, and 700 Exabytes in 2009. Data and content is expected to grow to over 35 Zettabytes by 2020, adding up to a 40x increase in a decade. For example, Microsoft Update and Windows Update push out a Petabyte of updates monthly. Cisco predicts that by 2013 annual internet traffic flowing will reach 667 Exabytes. A social network like Facebook produces 10TB/day of data, with Tweeter is not far behind (7 TB/day); each of the 4.6B mobile phones produce several events per seconds which need to be stored, processed analyzed and also the 30B RFID tags collectively generate a large data amount to be processed. Likewise, the 2B Internet users also generate a variety of events that can have important value in areas like statistics, demographics or marketing. And the 50B connected devices expected by the year 2020 will cause all of the previously mentioned figures to balloon even further. Domains like gaming and virtual worlds are also turning into a massive data management problems, with companies such as Zynga processing over 3TB/day of data for their 300M users.

In the scientific community, discovery has turned into a data-driven process, which represents a relatively new fourth paradigm in science [HTT09], next to the empirical, theoretical and computational models. The problem is that, at today's projected computing speed, it might take over a decade to gain some understanding of just what has already been archived from the most important scientific experiments. All fields of science (astronomy, physics, energy, medicine, drug discovery, climate, public health, etc.) are completely swamped with data today and major breakthroughs will be needed in repositories, storage, and computing architectures – all of which are central to the HiPEAC mission.

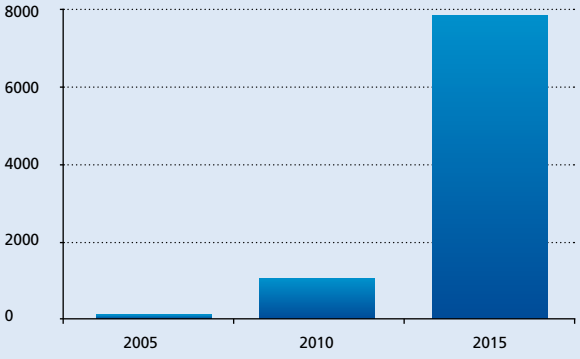
If we compare this growth with Moore's law (transistor density doubling every two years, see below), it is clear that data is on a higher exponential growth curve than computation capacity, and this unprecedented trend is forcing us to reevaluate how we work with data in computer systems.



Data growth vs. Moore's Law trends in the last 5 years. Data "deluge" means that we are heading towards a world where we will have more data available than we can process.

By 2020 the world will generate 50 times the amount of data and 75 times the number of "information containers" while IT staff to manage it will grow less than 1.5 times, and Moore's law predicts an increase in the number of transistors of about 25 over the same period. Hence, the amount of data outpaces the growth in processing capacity.

A Decade of Digital Universe Growth
Storage in Exabytes



Source: IDC's Digital Universe Study, sponsored by EMC, June 2011

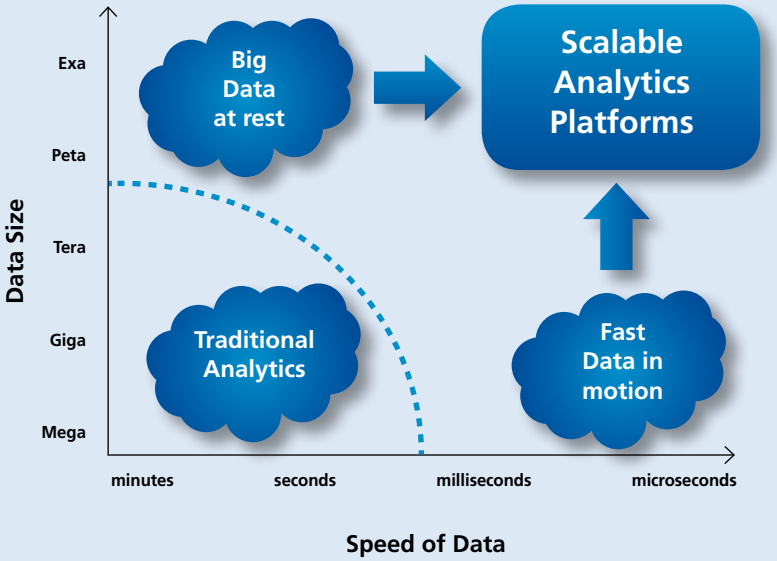
The IDC study predicts that overall data will grow by 50 times by 2020, and unstructured information — such as audio, email and video — will account for 90% of all data created over the next decade.

To provide value, systems will need to autonomously and intelligently extract structure and context, present it in an intuitive and interactive manner, and do so in near-real time. As a result, we will see a phenomenal increase in the required computational capabilities and scalability over today's systems.
Furthermore, the amount of unstructured data will increase dramatically. Physical data, in the form of free text, audio, video, images, location, sensors records, and scientific surveys and simulations, will dominate. This data differs from structured and semi-structured (machine generated) textual data in that it is far larger, noisy and imprecise in nature, more difficult to extract information from it, and also far more (thousands of times) complex to process. These requirements of structured interpretation, rapid visualization, and the need to process physical data, will require a vastly increasing demand for scaling the computational capabilities, and a new approach to define where this computation is performed.

One important set of applications deals with extracting *deep insights on data at rest*. These include traditional searches and analysis of scientific data, as well as more novel areas such as sentiment analysis through unstructured social network data, or natural language queries of large repositories. The computing aspects of these applications are challenging, and require massive processing and storage capabilities similar to today's supercomputers, but at far lower energy and cost. The "data at rest" already starts exceeding the capacity of near-line storage (e.g. the historical data collected by the largest Internet search engines). Additional challenges come from the requirement of bringing the relevant "cold" data, archived in off-line storage, to computing elements that can process it and extract the required insights. We call this the challenge of "warming cold data", and we believe this will represent an important aspect of the future data deluge challenges.
The other emerging set of applications deals with extracting *fast insights on data in motion*. This is a relatively recent field (of "real-time analytics") where decisions need to be taken quickly, often in a distributed manner, and with imperfect and partial data. Quick analyses are key to previously mentioned areas such as financial trading and safety systems, but also for delivering a compelling customer experience (by rapidly identifying trends and providing location-aware instant information), for security (surveillance, network perimeter analysis, fraud detection in electronic transactions), for public health (identifying pandemics), and for optimizing transportation. For example, since all our digital systems are continuously under attack, the ability to analyze data streams of logging information, identify anomalous behaviors and patterns, and rapidly reacting to them will be an important application of *fast data* analysis.

One of the computing systems challenges for the next decade will be coping with this increase in data and the simultaneous demand for faster processing. This will require advances in the state of the art of distributed computing, ultra low power col-

Classification of the Big-Fast-Total Data Space



We can classify 'big-fast-total' data along some important dimensions: size, generation and decision frequency, and structure.

Cognitive Computing on a Chip

…called cognitive computers, systems built with these chips won't be programmed the same way traditional computers are today. Rather, cognitive computers are expected to learn through experiences, find correlations, create hypotheses, and remember – and learn from – the outcomes, mimicking the brains structural and synaptic plasticity. …

"This is a major initiative to move beyond the von Neumann paradigm that has been ruling computer architecture for more than half a century," said Dharmendra Modha, project leader for IBM Research. "**Future applications of computing will increasingly demand functionality that is not efficiently delivered by the traditional architecture.** These chips are another significant step in the evolution of computers from calculators to learning systems, signaling the beginning of a new generation of computers and their applications in business, science and government."

…

IBM's overarching cognitive computing architecture is an on-chip network of light-weight cores, creating a single integrated system of hardware and software. This architecture represents a critical shift away from traditional von Neumann computing to a potentially more power-efficient architecture that has no programming, integrates memory with processor, and mimics the brain's event-driven, distributed and parallel processing.

…

Future chips will be able to ingest information from complex, real-world environments through multiple sensory modes and act through multiple motor modes in a coordinated, context-dependent manner.

(from http://www-03.ibm.com/press/us/en/pressrelease/35251.wss)

located sensing and computing, distributed analytics algorithms, tools and methodologies to identify the optimal computing location for data streams and the associated analysis and decisions tasks.

Much inter-disciplinary research is needed in this area: novel analytics platforms that can consume and analyze rich media data (sensor data, video, structured and unstructured data); novel algorithms and operators to process and visualize the big-fast-total data sources; novel data stores that can scale in performance and geographic distribution, including in the cloud; novel co-designed hardware that can leverage new disruptive technologies. Dealing with the data deluge will have far-reaching consequences, including the way in which we design the processor and memory hierarchy, which will have to be revolutionized to address the new data- and energy-imposed bottlenecks. For example, new data-centric approaches might emerge [P11], where every piece of data will come with its own built-in compute capabilities. This will require breakthroughs in the main HiPEAC research topics: architecture, programming and compilation.

To provide value, systems will need to autonomously and intelligently extract structure and context, present it in an intuitive and interactive manner, and do so in near-real time (even faster if one wants to search archives). Natural language processing and image analysis and recognition are clear challenges for the coming years.

Processing this kind of data requires the development and use of more sophisticated and more intelligent algorithms (e.g. recognition, mining and synthesis – **RMS**, as coined by Pradeed Dubey at Intel [Dubey2005]). All these algorithms are very compute intensive. As a result, we will see a phenomenal increase in the required computational capabilities, and a dramatic need to increase in workload predictability and scalability over today's systems.

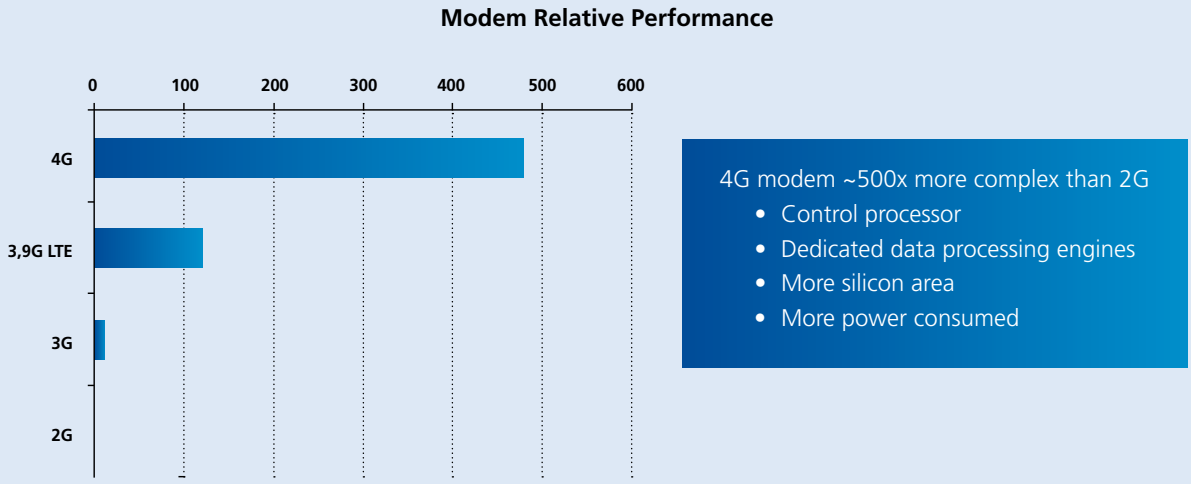
A side effect of most applications using natural data is that time becomes a first class citizen in the processing. The computing system cannot stop the real world or ask it to repeat an answer. Processing natural data also often occurs in a scenario where latency is important (example: ABS brakes, cyber-physical systems, augmented reality, path navigation, etc.)

Computing Systems Challenges: develop new computing paradigms and architectures to efficiently process natural data. The timing characteristics should be explicitly handled.

Ubiquitous connectivity and functional convergence: In the near future every electronic device will be connected. This reality is already appearing in the form of Internet-connected phones, airplanes, and street signs, but in the next decade such connectivity will become ubiquitous for everything from refrigerators to pacemakers. The interconnection of devices that interact with the physical world will create ubiquitous, large-scale cyber-physical systems. Such an interconnected world will enable the commoditization of computation, wherein devices can optimize where and when they store and process data to minimize cost and energy. This commoditization will go well beyond the cloud computing we have today to provide much greater opportunity for efficiency.

For example, a mobile phone may decide to send part of its

Chip Complexity Skyrocketing



As speeds increase, the chip complexity needed to achieve them skyrockets.

"I don't think the future is going to be quite like the past," "There may be trouble ahead."

"2G, back in the early 90s, was a hard problem. It was solved with a general-purpose processor, DSP, and a bit of control logic, but essentially it was a programmable thing. It was hard then – but by today's standards that was a complete walk in the park."
"A 4G modem which is going to deliver about 100X the bandwidth ... is going to be about 500 times more complex than a 2G solution."

"The 4G-modem problem will be solved by throwing a ton of dedicated DSP processing engines at it – which will, of course, require a lot of silicon real estate."
"But that's not so bad, because silicon is being scaled the whole time. But it's going to eat a lot of power, and power is the real problem."

Simon Segars, EVP and head of ARM's Physical IP Division, during his keynote at the Hot Chips conference 2011.
(from http://www.theregister.co.uk/2011/08/20/microprocessors_may_face_trouble_ahead/)

video processing to a cloud computing facility based on the power and costs of transmission and local and remote computation. Such a decision might change depending on the time of day (cost of computation), the distance to the cell phone base station (cost of communication), and the remaining battery life (available energy). Such an interconnected world is not confined to computing devices only, but will interact and provide services and resources for other highly complex systems such as cars, aircraft, and infrastructure systems.

An interconnected world also opens up a wide range of security and dependability issues [Avizienis2004]. The massive amounts of data available online must be secured and the ability to use the online resources must be ensured. Access to something as simple as a refrigerator may not seem like a large security risk, but the ability to bring down a whole power grid by suddenly shutting down millions of them is. With ubiquitous connectivity comes the requirement to provide security and authentication that can enable commodity computing without risking proprietary data.

With *functional convergence* consumers expect all devices to share more and more functionality. They expect to have access to their online and personal information, as well as communications, entertainment, news, and social networks on all devices, from their TVs to phones and computers. Networked Internet access has become the exclusive form of data exchange, with telecom operators, broadcast networks, and Internet service providers fighting to deliver voice, data, and entertainment to consumers over the same copper wires, fiber optic cables, or wireless networks. They not only want their data to be efficiently accessed, but also their applications to be accessible everywhere. This leads to virtualization solutions allowing migrating programs from one platform to another, even if the latter platform does not necessarily offer completely equivalent functionality and features.

Computing Systems Challenges: security and authentication for large ad-hoc networks of devices; transparent movement of computation and data without losing confidentiality. Interoperability of systems, efficient communication infrastructures. Virtualization solutions allowing a smooth migration from hardware to hardware.

A.4. New technological challenges and opportunities

Many obstacles will limit the scalability of CMOS in the next decade: power density, quantum effects, manufacturing variability, manufacturing costs, and design productivity. It is clear that we will not be able to use all the transistors on a chip at a time due to power dissipation constraints. Caches, out of order execution, branch predictors, etc., have been needed for years already to keep up the stored program illusion. We also see a rising variety of processor architecture with multi-cores, GPUs, FPGAs, DSPs, and combinations of those using various interconnection network topologies and more or less complex memory architectures.

However, a major paradigm shift is taking place now. "Moore's law", while keeping its pace on transistor density, will only allow a minor increase of frequency and decrease of power dissipation per transistor. Even if it will still be feasible to pack more devices on a chip, the power dissipation of each device will not be reduced accordingly. Since we are already pushing the limit of power dissipation/consumption, it will not be possible anymore to use all transistors on a chip simultaneously. New technology nodes also add more leakage power, more variability and less reliability.

Power wall: The performance of computer systems is defined by their available power. For high-end systems the available power, and hence performance, is limited by the ability to cool the machine and the processor. This limitation is so severe that chips in the next five years will only be able to activate 75% of their circuits at any given time without overheating. For portable devices, performance is limited by the maximum temperature of the device and by the battery capacity. For all systems, higher power efficiency directly leads to higher performance. As a result, power efficiency is the leading constraint in hardware design, and is being felt in software design as well.

In addition to performance, power defines the cost of the systems. For high-end machines, the cost of power and cooling equals the purchase price of the hardware after only four years. For portable devices, power requirements determine the size of the battery. This efficiency demand will drive new computer architectures and designs that trade-off absolute performance for better efficiency.

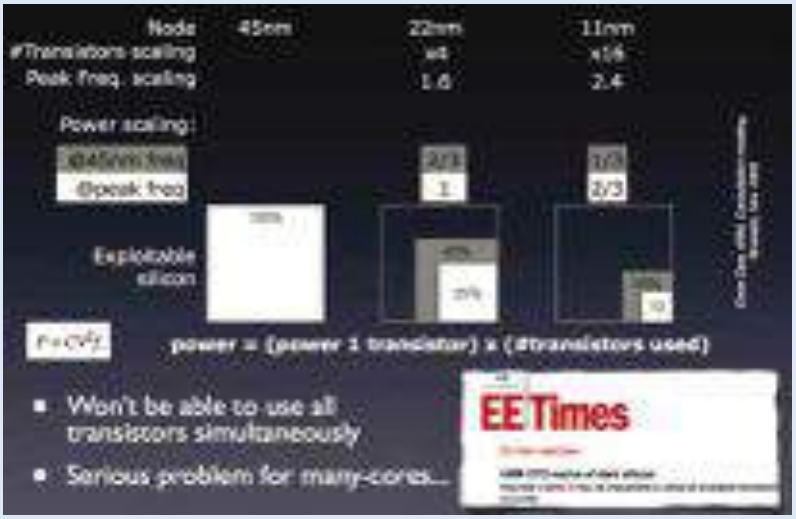
Computing Systems Challenges: energy efficiency above all else; enabling energy-efficient software; heterogeneous architectures and programming systems; hardware-software interaction to control energy consumption.

Higher energy efficiency and delivering reliable behavior from unreliable and highly disperse components requires investigating new research directions at all levels.

It is critical to highlight and anticipate the emerging technologies that will shape the future of computing systems in the context of the trends and analyze the impact such technologies can

Dark Silicon

We are entering the era of Dark Silicon. Features sizes will continue to decrease for the next decade but power scaling has largely stopped, meaning that the increase in the number of transistor per chip is not offset by a similar decrease in power consumption per transistor. This has resulted in an inability to turn on all of a processor without exceeding its power budget (power wall). As a result, the hardware and software will have to jointly optimize which parts of the chip will receive how much power to meet performance and efficiency goals.

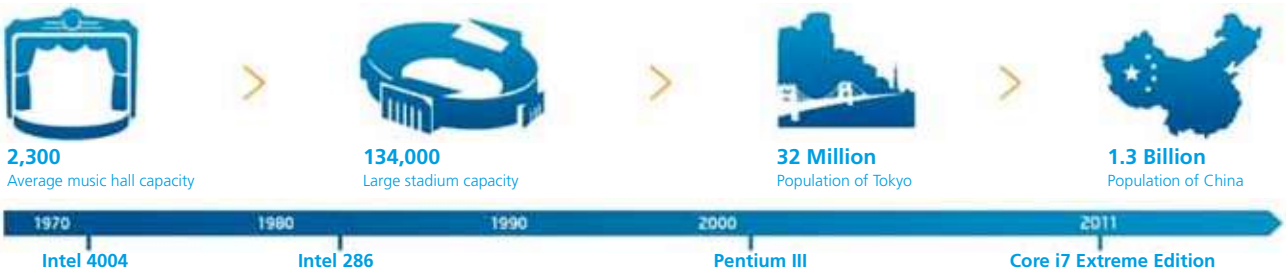


Dark Silicon (From ARM) [Aitken2011], [Esmaeilzadeh2011]

VISUALIZING PROGRESS

If transistors were people

If the transistors in a microprocessor were represented by people, the following timeline gives an idea of the pace of Moore's Law.



Now imagine that those 1.3 billion people could fit onstage in the original music hall. That's the scale of Moore's Law

Illustration of Moore's law, according to Intel [Intel]

have on our computing systems. Disruptive technologies like silicon photonics, non-volatile memories, and probabilistic transistors may represent decisive opportunities in terms of efficiency, dependability and, in some cases, reduced design complexity. However, such technologies will drastically impact the way we design our systems as well as the tools to program them in order to exploit their advantages.

Studying the impact and maturity of these technologies as well as the positioning of Europe in terms of industrial advantage is also a major issue the computing systems community has to investigate in order to have a competitive advantage.

New computing elements: As new technology emerges, several possible paths for alternative architectures emerge as well:

- ***Probabilistic CMOS.*** This approach, pioneered by K. Palem [Palem05], consists of using standard CMOS, but lowering the voltage of the transistors. As a consequence, the energy is significantly reduced but the transistor provides the correct output only with a certain probability. However, large classes of important algorithms (e.g., optimization algorithms) are compatible with such a probabilistic computing medium. First prototypes show very promising energy gains.
- ***Memristors.*** These are passive two-terminal devices of which the resistance is roughly proportional to the charge flowing through the device. They are only recently implemented in a dense manner [Strukov2008], and can first be used to implement very large, very dense, low-power reconfigurable arrays, which opens the door to low-energy, low-area, reconfigurable co-processors. Memristors are also ideally suited for the hardware implementation of synapses in hardware neural networks. As high-performance applications are increasingly about RMS, the application scope of neural networks becomes very significant. More importantly, as the number of faulty components increases, hardware neural networks provide accelerators that are intrinsically capable of tolerating defects, without identifying or disabling faulty components, simply by retraining the network.
- ***Neuromorphic computing elements.*** Gaining inspiration from the brain is one way to improve our computing devices and to progress beyond Moore's law. Since the introduction of the memristor, it has been recognized for its use as a synapse-like element and the possibility for a direct implementation of the STDP (Spike-Timing-Dependent Plasticity) learning rule [Snider2008]. Similarly, organic devices have been demonstrated to have synaptic properties [Alibart2010] and recently phase change memory was proposed for the same purpose [Kuzum2011]. Beyond the fact that the STDP learning strategy implies a paradigm shift in information coding (from state based to event based), it promises easier implementation of very dense arrays and more energy efficient systems for qualified classes of applications (e.g. sensory data processing, vision). Big research initiatives in the field of neuromorphic computing are currently under way, such as the DARPA funded SYNAPSE project coordinated by IBM and in Europe several computing projects with FP6-FACETS/FP7-Brainscale being the most notorious.
- ***Graphene.*** Other potentially disruptive technologies are emerging, such as graphene (Physics Nobel Prize in 2010) transistors, which seem capable of increasing clock frequency beyond the capabilities of silicon transistors. Currently such transistors are significantly bigger than silicon transistors, and only limited circuits have been implemented. Their application scope is mainly fast and low-power analog signal processing, but research on graphene transistors is still in its infancy, and this technology should be carefully monitored by the computing industry.

New interconnects: 3D stacking offers a new freedom for interconnecting devices, allowing using the third dimension by stacking dies (of potentially different technologies). Sev-

eral techniques exist (copper-to-copper interconnect, through silicon via, optical) that allow high-bandwidth interconnect between dies. It adds a new interconnection dimension to the SIP (System in Package) approach already used in products: 3D-stacking allow to exchange data by connections on all the surface of the die, and SIPs only allows connections on sides of the dies. This technology allows the rethinking of architectures by physically reducing the distance between memories and processing (for many-core architectures), or between sensors and processing (intelligent retinas). Silicon photonics is another technology that could have an impact on realizing efficient architectures.

Self-building nanostructures: building systems at the nanoscale will be a challenge because technologies to manipulate elements of this size are only in research labs now. Two paths can be followed:

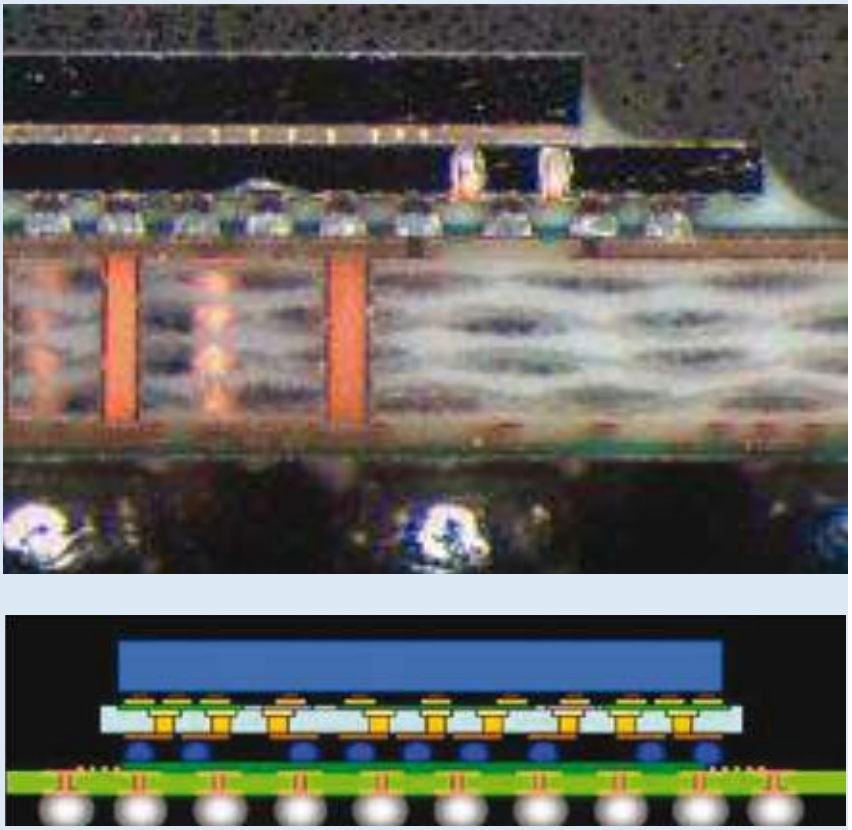
- Self-building and assembling devices: this can be done, for example, using DNA. The DNA hybridization (the attraction between bases and a specific chemical, a property of the end of a DNA strand), will enable building complex nanostructures.
- Directed assembly of nanoscale computing engines: DNA strands can be used to manipulate nano-objects, such as carbon nanotubes [DNA].

One example of nano-architecture can be found in [Pistol2010]. The simplicity of the compute unit, their easy duplication, could lead to a rebirth of massive simple cores with a serial architecture, similar to the Connection Machine CM1 [CM-1].

More-than-Moore: The result of Moore's Law has driven the semiconductor industry for decades, pushing technology towards extremely fast processors, huge memory sizes and increasing communication bandwidth. During those decades, ever more demanding applications exploited these growing resources almost as soon as they arrived on the market. A major paradigm shift is taking place now, however, both in the technology push and in the application pull. The result of this paradigm shift has been called the "More than Moore" era by many authors; see for example [MtM].

3D Stacking

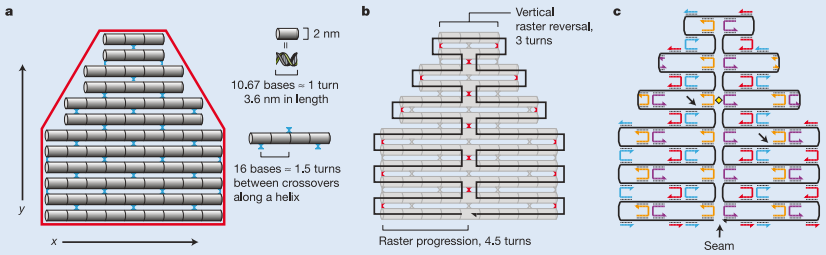
The advent of 3D stacking enables higher levels of integration and reduced costs for off-chip communications. The overall complexity is managed due to the separation in different dies, independently designed.

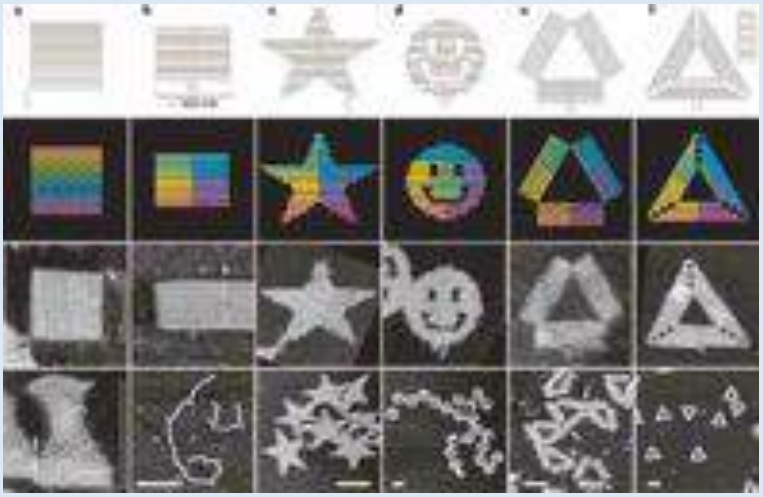


Photos: STMicroelectronics & CEA-LETI:
Multi-die stacking using Copper-pillars and TSVs

Self-building Nanostructures: DNA Origami

DNA Origami: DNA as a structural material





From Architectures for Emerging Nanotechnologies, Alvin Lebeck, ACACES 2011 summer school, originally from [Rothemund2006]

From the point of view of the technology push, two observations have to be made. First, the cost levels for system-on-chip development in advanced CMOS technology are going through the roof, due to increased fabrication, design, and verification costs. Secondly, the continuing miniaturization will end Moore's Law for silicon one day in the not so distant future.

From the application pull perspective, it has become clear that consumers and society have by and large lost interest in new generations of devices that only feature more computational power than their previous generation. For improving the consumer experience radically new devices are needed that are more closely integrated in everyday life. These devices will require sensors, mechatronics, analog and mixed-signal electronics, and ultra-low-power and/or high- performance technologies to be integrated with CMOS technology to directly drive actuators, light devices or interfaces with the world.

Devices that embed multiple technologies are instances of the "More than Moore" approach: combining generic CMOS-technology with new technologies for building more innovative, dedicated, smarter and customer-tailored solutions. This new era of systems will certainly trigger innovation, including new methodologies for architecting, modelling, designing, characterizing, and collaborating between the domains required for the various technologies combined such a system. Further, tight integration will provide for reduced cost and greater functionality.

The "Moore's Law" race towards ever-larger numbers of transistors per chip and the "More than Moore" trend to integrate multiple technologies on silicon are complementary to achieve common goals such as application-driven solutions, better system integration, cost optimization, and reduced time to market. Some companies will continue to follow the "Moore's Law" approach, while others will shift towards the "More than Moore" approach. This will drive industry into a direction of more diversity and wider ecosystems.

B. SWOT Analysis of Computing Systems in Europe

B.1. Strengths

The European computing systems industry has a strong embedded ecosystem spanning the entire spectrum from low power VLSI technologies to consumer products. Companies such as ARM, STMicroelectronics and ST-Ericsson are leaders in providing semiconductor processing elements and IP for embedded systems. Large end-user European companies have a strong market presence internationally in areas such as automotive (Volkswagen, Renault-Nissan, Peugeot-Citroën, Fiat, Daimler), aerospace and defense (Airbus, Dassault, Thales, Saab), and telecommunications infrastructure (Nokia, Ericsson). These companies are all globally competitive and work on the forefront of embedded system design and implementation.

These larger players also rely on a thriving community of SMEs that strengthen the technical and innovative offers in the market. This strong embedded ecosystem creates a fertile environment for innovation, better integration and worldwide leadership.

From an educational perspective 204 European universities are rated among the top 500 universities in the Shanghai Jiao Tong University ranking [ARWU2011]. This is more than the United States of America (154 universities). The European university system thus benefits from a very strong educational environment and a highly competitive undergraduate and graduate educational system. The ongoing bachelor-master transformation will further strengthen the European educational system.

Europe benefits from a large pan-European centralized research program through the Framework Programmes for Research and Technological Development. If used properly, these serve as a powerful tool to direct community-wide research agendas and a strong incentive for diverse groups of companies and research institutions to work together across political and cultural divides.

B.2. Weaknesses

European computing systems research is characterized by a weak link between academia and industry, especially at the graduate level. Companies in the United States value PhD degrees much more than European companies, which often favor newly graduated engineers over PhD graduates. This leads to a brain drain of excellent computing systems researchers and PhDs trained in Europe to other countries where their skills are more valued. As a consequence, some of the successful research conducted in Europe ends up in non-EU products or does not make it into a product at all.

The End of Silicon Scaling

"Silicon scaling will end at some point, and I think **it's coming sooner** than many people think," "you can only scale so far before we need other materials like III-V semiconductors,"
"You need to produce 200-300 wafers an hour, and today's EUV machines can do about five wafers per hour now,"
"Some people question whether it ever will be mainstream-lots of R&D still needs to go into it,"

Simon Segars, general manager of ARM's physical IP division in a keynote at the 2011l Hot Chips event

(from http://www.eetimes.com/electronics-news/4218909/ARM-wrestles-with-silicon--battery-hurdles)

The lack of a venture capitalist culture contributes to the brain drain. It is much harder for a university or PhD graduate to start a company in Europe than in the United States. Yet even with venture capital, the personal investment and identification of startup employees with the fate of their company attitude found in Silicon Valley startups is largely absent from the European work ethos and organized labor agreements. On top of this, bureaucracy and administrative procedures in some countries are preventing or killing several new initiatives.

From an industrial point of view, Europe lacks highly visible pan-European players in the computing systems domain, especially compared to the USA. This severely reduces the potential synergies, impact, and large-scale development capabilities of these industries. In particular, Europe lacks a major player in the world of high-performance computing such as HP, Cray, Intel, IBM, or NVIDIA in the USA. Computer components, such as microprocessors, GPUs, and memories, are all produced, and more critically, designed, outside Europe, with the notable exception of ARM. On the regulatory side of the picture, the inertia caused by administrative overhead and IP regulations, significantly hamper the industry.

At the research level, European research in computing systems is lacking international and community-level visibility due to the absence of a sufficient number of highly visible computer engineering departments. This leads directly to many of the best PhD candidates deciding to pursue studies, and often careers, abroad in the United States.

The lack of open source tools in the computing systems domain (for example synthesis tools) is a weakness for European research in the computing domain. This is less of a problem in the United States because universities typically have access to free or low-cost licenses for commercial tools, through personal contacts enabled by closer physical proximity. This prevents small groups, start-ups, and universities from having a significant contribution to the innovation in the hardware domain. Existing open source CAD tools are not generally usable, commercial emulation platforms are expensive and not readily available, and testing ideas on real silicon is still a marathon that requires solid financial backing and extensive experience, neither of which are readily forthcoming in Europe.

It is also worth noting that the language and cultural diversity in Europe are handicaps to attracting bright international students to graduate programs outside of their home country. Lack of command of English by graduates in some countries also greatly hampers international networking, collaboration, and publication.

B.3. Opportunities

Paradoxical as it may seem, several challenges that society is facing are at the same time also huge opportunities for the research and industry in computing systems. For example, the aging population challenge will require the development of integrated health management and support systems that enable people to live at home for a longer time. Europe's national health systems are far better placed to take advantage and coordinate such efforts than that of the United States or China. European expertise in low-power and embedded systems and its SME ecosystem is an asset for tackling other grand challenges such as environment, energy and mobility. Experience in mission critical systems gives Europe a competitive advantage in the safety and security challenges ahead in larger scale consumer systems.

Convergence leads to new business opportunities, especially due to ARM entering the personal computing and the data center fields. New opportunities for ARM-centric software solutions in these new domains will appear and the European industry will be well positioned to exploit them. De-verticalization of the market is good for niche players who can then compete with the dominant players thanks to the availability of commodity components. Collaborative development enables more reuse and stimulates investing resources in opening niche markets that would otherwise be too unprofitable to enter. For example free software projects such as Linux or GCC create communities and enable start-up companies to enter the market without being dependent of major US companies like Microsoft.

Disruptive technologies such as cloud computing and the convergence of HPC and embedded computing represent opportunities for Europe too. The trend towards more distributed environmentally integrated cyber-physical systems could be beneficial to the European semiconductor industry, which has significant expertise in the wide range of required technologies.

The recent move to classify micro- and nano-electronics as key enabling technologies [KET] for Europe creates significant opportunities for the computing systems industry in Europe. This move comes in response to Europe's semiconductor market share decreasing from 21% to 16% since 2000 [KET]. Such a large and centralized research and development planning presents a great opportunity for computing systems research in Europe, bringing it on par with other technologies such as energy, aerospace and automotive technology. The resources available for developing pan-European research capabilities could be used to address several of the weaknesses addressed above, in particular the lack of tools and hardware development expertise. The emphasis on collaboration with industry, in particular SMEs, can provide mentoring and bridges for researchers wishing to industrialize their results, if used properly. And most importantly, the large amounts of money allow the Commission to target critical areas.

Europe's cultural diversity creates opportunities for Europe in a global world that will not be dominated primarily by Western companies and institutions anymore. European companies are more sensitive to cultural differences, which may become important in developing new markets all over the world.

Finally it is worth noting that the proximity of Europe to the Middle East, the Russian Federation and Africa represents a huge market opportunity and should not be neglected.

B.4. Threats

Currently most high-end and middle-end general-purpose processor technology is developed in the USA. China is also developing its own hardware, of which the Loongson processor is the best-known example. There is no reason to believe they will not become highly competitive in the low-end/embedded market within the next decade. With the development of low-power processors competing with ARM, Europe risks losing its dominance in the low power processor market.

The economic and financial problems in the Eurozone are impacting the investment climate and the consumer confidence. The draconic debt reduction measures that are required in some European countries inevitably also reduce the budgets for public research funding and education, weakening the foundations of a modern economy.

C. The HiPEAC Core Computing Systems Challenges

From the trends, it is possible to identify three main computing systems challenges: **improving efficiency, managing complexity,** and **improving dependability.**

C.1. Improving efficiency

Multiple performance metrics: In the past, compute performance drove the computer industry. Customers were happy to buy a faster computer and to pay the same price (Machrone's Law). When the introduction of new software functionality slowed down, new computers did not need to run users' software faster. The need for extra cycles leveled off and the customer's focus moved to price instead of performance. In the current economic climate cost constraints have become more critical than ever. In general purpose computing, *performance per Euro* is key.

For embedded devices, the criterion of choice is *performance per Watt and/or per Euro*. Applications must runs in a given budget despite a regular increase in computation demand, storage capabilities and inter-operability capabilities. In mobile systems the regular reduction in battery life that occurred in the past five years has reached its limit and battery operated devices must now sustain performance increases at a constant energy budget, to compensate the limited improvement in battery capacity.

Due to the rising operational costs of energy and cooling, and because chip packaging costs contribute significantly to the final costs of hot-running chips, the criterion of *performance per Watt per Euro* has also become key for the data center. As previously pointed out, more and more consumers prefer the best price for reasonable performance, rather than the best performance at all costs. Companies are looking to further reduce their computing infrastructure costs, potentially leading to new business models based on renting out computing power and storage space.

Power defines performance: Moore's law and the associated doubling of the number of transistors per IC every process generation, is no longer accompanied by a sufficient reduction in voltage to compensate for the current increase. Such voltage reduction is becoming less effective because we are approaching the limits of threshold voltage. For static power, the leakage current is also increasing drastically in the new technology nodes. The current approach to master power consumption is to control the voltage and the frequency of the cores: it is called DVFS (Dynamic Voltage Frequency Scaling). At the same time, the ITRS projects that device shrinkage will continue for at least another few generations [ITRS]. Therefore, while future chips are likely to feature many more transistors, only a fraction of the chip will likely be active at any given time to maintain a reasonable power limit (see the earlier discussion of "Dark Silicon"). This will apply to servers, desktops, laptops and embedded devices. Hence power consumption will always dictate how much compute performance you can obtain from a device, either because of thermal limitations, or due to battery capacity.

The Limit of Air Cooling

The limit to air cooling is 130 W for a chip. That limit was reached around 2004.
Committee on Sustaining Growth in Computing Performance; National Research Council, The Future of Computing Performance: Game Over or Next Level? The National Academies Press, 2011.

Available online at: http://www.nap.edu/catalog.php?record_id=12980.

Taking a Disruptive Approach to Exascale

"It's not about the FLOPs any longer, it's about data movement. And further, it's not simply a matter of power efficiency as we traditionally think about, it's about locality.
…
Algorithms should be designed to perform more work per unit data movement
…
programming systems should further optimize this data movement."

Bill Dally, Chief Scientist, NVIDIA
(from http://www.hpcwire.com/hpcwire/2011-08-18/taking_a_disruptive_approach_to_exascale.html?featured=top)

Communication defines performance: Communication and computation go hand in hand. Communication — or, in other words, data transfers — is essential at three levels: between a processor and its memory, among multiple processors in a system, and between processing systems and input/output (I/O) devices to the outside world. As transistors and processors become smaller, the relative distance of communication increases, and hence so does its relative cost.

Current computer systems generally use shared memory, i.e. where from a software point of view all memory is reachable from any processor in a multi-processor system. The other

extreme is message passing, where all memories are private and explicit communication is required to exchange data. Providing the illusion of shared memory across ever increasing numbers of processors is a costly endeavor in terms of power and performance. However, developing software to use explicit communication through message passing is far more expensive in terms of development cost. To move forward we need tools to simplify development of software using explicit communications to enable simpler, more efficient hardware.

Heterogeneity and accelerators to the rescue: Specialization of the hardware is one of the key aspects to improve performance and efficiency. Since it will not be possible to use all cores at once due to power constraints, it makes little sense to make them all identical. Hence, it becomes important to specialize the cores for specific tasks. As a result, the functional and micro-architectural heterogeneity that is seen today in embedded systems is becoming a direction for general purpose computation to meet its demands in terms of performance and power consumption. For example, Intel's TCP/IP processor is two orders of magnitude more power-efficient when running a TCP/IP stack at the same performance as a Pentium-based processor [Borkar2004]. This additional power efficiency enables taking full advantage of the additional transistors that become available thanks to Moore's Law as soon as the workload can be efficiently spread across multiple heterogeneous resources.

Unfortunately, the non-recurring engineering (NRE) costs of complex application-specific cores and ASICs are rising dramatically. This evolution is primarily caused by the climbing costs of creating masks for new manufacturing technologies and the cost of verifying increasingly complex designs. The ESIA 2008 Competitiveness Report [ESIA2008] illustrates this trend. In addition to the cost of managing the complexity of the design itself, verification and validation are becoming prohibitively expensive. Finally, the integration and software development costs also have to be taken into account. As a result, chip development costs can only be recovered by selling large quantities of ASICs. Unfortunately, ASICs are, by definition, application-specific, and are often tuned to the requirements of a few big customers. Therefore, they cannot be used "as is" for large numbers of applications or customers. These economics make it impractical to build ASICs for most applications unless newer technologies drastically reduce development and verification costs, or unless the lifetime, power consumption or reuse of the ASICs overcome the cost. ASIPs (Application-Specific Instruction-set Processors) are domain specific and mitigate the efficiency of ASICs with the larger use of general processors. The frequency limit of programmable cores makes ASICs more competitive though.

Message passing vs Shared Memory

"It's very distressing - I'm watching almost with disbelief. The Americans cannot get it out of their heads that if you're trying to build machines with lots of processors, you don't assume that they all share a common memory. **The world doesn't have a common database. We pass messages to one another."**

David May, professor of computer science at the University of Bristol, (and the architect of the Transputer in the 80's), talking about the current trend in chip design that proliferates cores - Intel's 'Knights Corner' currently runs to 50 processors on a single chip - but has them all dipping into the same memory pool.

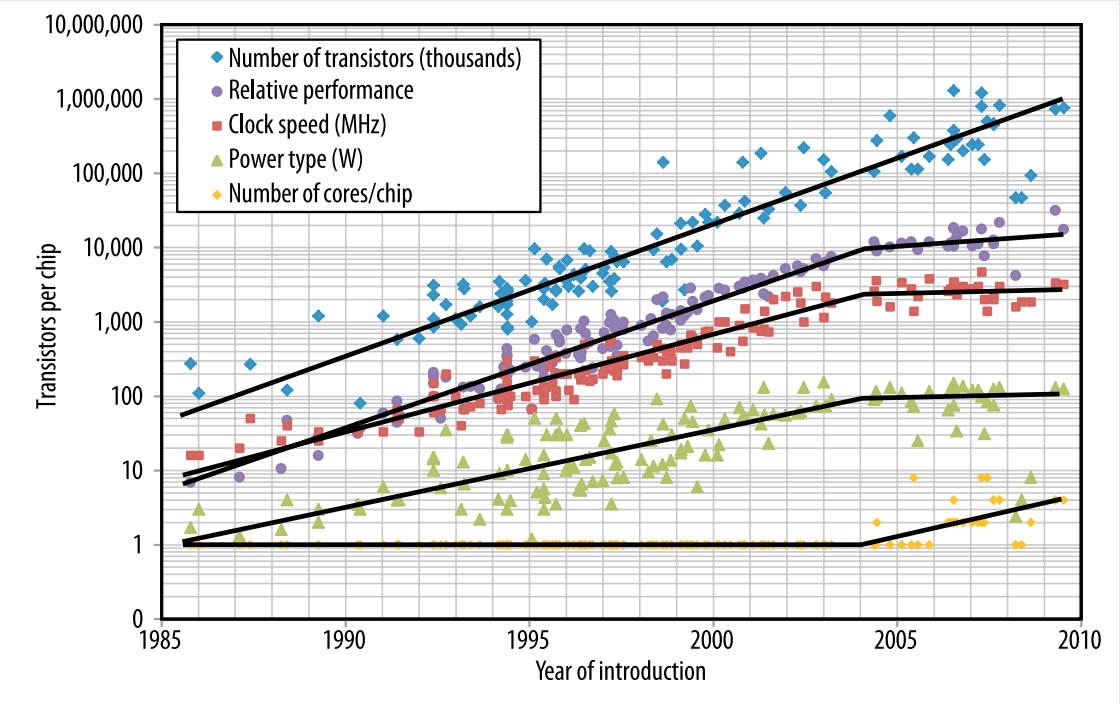
(from http://www.reghardware.com/2011/08/18/heroes_of_tech_david_may/)

The complete stack is back: Cross layer optimization. The market now largely depends on de-verticalization and specialization of the different layers of the product stack, ranging from software down to hardware manufacturer. We observe, however, that a few key companies, such Apple, Google (who recently acquired Motorola Mobile), are actively seeking to control and to implement the whole stack from hardware through to the end-user experience. They do this to ensure the quality of the user experience and to minimize cost. Controlling and implementing the whole stack allows them to efficiently optimize across all layers and avoid the inefficiencies introduced by the need to standardize at each layer. Therefore, for de-verticalization to remain a viable option for efficient and competitive products, innovative methods and tools for cross layer optimization need to be investigated.

C.2. Managing complexity

The reign of legacy code: Hardware evolves faster than software. It is a consequence of the fact that the economic lifetime of software is much longer than the economic lifetime of hardware. Rather than looking for software to run on a given hardware platform, end users are now looking for hardware that can run their existing and extremely complex software systems. Virtualization has provided a new lease on life for legacy systems by enabling full system portability. Porting software to a completely new hardware platform is prohibitively expensive and in some cases requires re-certification of the software.

Moore's Law: limited by Power



Data from Kunle Olukotun, Lance Hammond, Herb Sutter, Burton Smith, Chris Batten, and Krste Asanovic [Fuller2011]

While the increase in transistor density continues (in blue) the maximum frequency has reached a limit by 2004 (in red). This triggered the increase of cores (in yellow) to keep performance increase (in purple). However, the power also reached a limit (in green), leading to new challenges [DarkSilicon].

At the same time, hardware is evolving at an unprecedented pace. The number of cores and instruction set extensions increases with every new generation, requiring changes in the software to effectively exploit the new features. Only the latest software is able to take full advantage of the latest hardware improvements, while older software benefits much less from them and customers that need more performance have to port and tune their applications to the latest systems. Given the rate at which new systems enter the market, this becomes an almost continuous process.

Parallelism seems to be too complex for humans: Programming parallel applications with basic concurrency primitives, be it on shared or distributed memory models, breaks all rules of software composition. This leads to non-determinism, unexplained performance issues, debugging and testing nightmares, and does not allow for architectural optimizations. Even specialists struggle to comprehend the behavior of parallel systems with formal models and dynamic analysis tools. Alternative concurrency primitives, such as transactional memory, suffer from other problems such as immaturity and a lack of scalability.

Most programmers should not be required to directly care about the details of parallelism, but should merely have to specify the partitioning of their sub-problems into independent tasks, along with their causal relations. Composable formalisms and language abstractions that offer exactly this functionality already exist. Some of these techniques are very expressive; some lead to inefficiencies in mapping the exposed concurrency to individual targets. There are huge challenges and difficult tradeoffs to be explored in the design of such abstractions, and in the associated architectures, compilation, and run-time support to make them scalable and efficient.

Frequency Wall makes ASICs more Competitive

As it happens, the shift to multi-core processing is creating new opportunities for specialized accelerators. In the past, general-purpose processor speed increased exponentially, so an ASIC would quickly lose its performance advantage. But this trend has slowed down considerably with clock speed leveling off. As a result, the performance benefits offered by ASICs can now be amortized over a longer period of time [Pfister2007]. This makes it more viable to optimize each core for its task. In heterogeneous multi-core designs, one way to efficiently use the transistors is to implement highly specialized accelerators (video, cryptography, DSP functions, etc.) and activate them on demand depending on the running application requirements. This additional power efficiency enables taking full advantage of the additional transistors that become available thanks to "Moore's law".

The heterogeneity of efficient hardware adds an extra layer of complexity for software developers. Effective software engineering practices cannot and should not let the programmers worry about the details of parallelism and its implementation on various resources. They should only focus on correctness and programmer productivity. Performance optimizations, including the exploitation of concurrency on a parallel or distributed platform, should be performed by automatic tools. David Patterson talks in this context about the productivity layer that is used by 90% of the programmers and the efficiency layer that is used by 10% of the programmers [Patterson2008].

Except for specific high-performance computation applications (where programmers are often experts in parallel computing) and for the design-space exploration of special-purpose systems, the quest for efficiency and scalability should never limit design productivity.

Hardware Complexity: The hardware has grown in complexity as well, but this complexity has been shielded from the programmers by a compiler and an operating system. However, this infrastructure is ill equipped to exploit many opportunities because the programming languages do not convey enough information about what the allowable or permissible operations are. With current computing systems including specialized accelerators, GPUs or reconfigurable processing units, SIMD extensions and energy management techniques, proper language constructs and careful design of runtime libraries and APIs are required to help exploit increasingly complex hardware. At the system level, virtualization techniques can help to hide the hardware complexity.

Efficiency vs Productivity: Energy vs Programming Efficiency

There was a similar dilemma in the past. In the late 80's and the 90's a similar battle was waged between shared memory and message passing. On one hand, proponents of message passing argued that it is far more efficient than shared memory, which added another complex layer on top of hardware messaging. Their opponents argued that while this is correct, shared memory is in fact far easier for programmers. Shared memory today is the prevailing paradigm for mainstream parallel computers. In hindsight the reason is obvious. While efficiency is very important, software development is the key factor for the success of the architecture. Even inefficient software is better than non-existing software.

While today we are concerned with power efficiency and the arguments definitely support heterogeneity, there is little disagreement that the heterogeneous architectures we have seen so far are hard to program. Homogeneity (perhaps in the form of a convergence of general-purpose cores and specialized cores) may offer a better path for software development. Thus, for heterogeneity to deliver on its promise of greater power efficiency it is critical that the system complexity and software challenges for heterogeneous systems are successfully addressed.

C.3. Improving dependability

Current chips for consumer applications are designed to run even in the worst-case scenario: at the lowest voltage, at the worst process technology corner and at the highest temperature. Chip binning, i.e. sorting chips after fabrication according to capabilities, is usually not performed because the testing costs outweigh the income from selling the chips. Microprocessors are an exception to this rule, as the selling price of these chips is far higher.

The practical upshot is that most consumer chips are over-designed. In most realistic cases typical use is far from the worst case, and this gap is widening as we move below 22nm due to process variability. The increasing complexity of SoCs also widens the gap due to the composition of margins. If the architecture and design methodologies do not change, we will eventually end up with such large overheads that it will become economically infeasible to produce new chips.

Worst-case design is not an option: ITRS predicts device feature sizes as low as 11nm by the end of this decade. These very small devices and the growing total device count in a system will significantly increase the variability in circuit behavior, and thus the probability of producing an incorrect result. The same observation holds for large supercomputers consisting of millions of cores. The mean time between failures (MTBF) can be as low as one day for today's systems. Further, future exascale class of systems will include order of magnitude larger memories. The current practice of check-pointing and roll-back will increase the pressure for orders of magnitude more disk/storage bandwidth [Kogge2008, page 150].

Even today's systems are vulnerable to such errors and the current de facto standard techniques for fault tolerance such as triple-modular-redundancy (TMR) [Spainhowver1999] and error correcting codes (ECC) are costly. TMR is simply too expensive in terms of die space and, more importantly, energy to include on all system components. The increased prevalence of ECC in consumer-level devices reduces their power efficiency. It is clear that in addition to technological breakthroughs in terms of bandwidth and alternative storage (for reliable and fast selective check-pointing) we need to develop design level and programming level approaches to maintain application productivity and correctness in the face of errors.

Errors can affect applications in three major ways. First, they can reduce performance, leading not only to longer execution times but also to higher energy consumption. Second, they can cause unexpected application aborts, which can cost large amounts of debugging effort looking for phantom coding errors and force applications to re-execute, wasting significantly more time and energy. Finally, and most importantly, they can corrupt the application's results, possibly leading to invalid conclusions derived from the results of the computation. These outcomes underscore the need to study the vulnerability of applications, identify their most vulnerable components and design techniques to reduce those vulnerabilities, or even develop programming paradigms and execution environment that no longer offer the illusion of reliable computation but rather allow developers to mitigate their effects.

New design methodologies and architectures will be required to cope with this problem. For example, the "Razor" concept [Ernst2004, Blaauw2008] is one solution. In this case errors are allowed to occur from time to time when typical conditions are not met, but they are detected and subsequently corrected. Alternative methods are using active feedback and quality of service assessments, or relying on local monitoring systems to tune the operating voltage and frequency to the real execution conditions. Some of the techniques currently under development however decrease the system's predictability while others lead to globally asynchronous systems. Therefore, performance predictability may be hard to achieve and guaranteeing real-time condition may still require over-provisioning of the design.

Systems must be built from unreliable components: Advanced device scaling is radically reducing the uniformity and reliability of wires and transistors. As a result, soft errors (radiation induced disturbances in memories and logic), hard errors (permanent functional degradation due to gate oxide wear-out, electromigration, and manufacturing defects), and variability (power and performance differences due to limitations in lithography) are increasing to the point where existing design methodologies become impractical. The solution is to address reliability at all levels of the system, from devices through architecture to operating systems and software.

Current design approaches provide the illusion of an externally fault-free device by over-designing and conservatively operating devices. Techniques such as error correction circuits, replacing defective memory arrays with redundant copies, and adding operating margins to ensure frequency and power requirements are becoming increasingly costly in terms of power and performance as variability increases and reliability decreases. To address this, we must move away from the illusion of fault-free devices and towards the goal of self-corrective reliable systems built out of unreliable components. This requires accepting the notion of unreliable hardware components at all levels of the system, and adapting device usage at runtime to account for hardware failures and behavior.

Time is relevant: Dependability does not only imply functional correctness, but also timing correctness. While timing is not a problem for many applications, it poses a major hurdle for systems that have to interact with the physical world. Examples are embedded systems, consumer systems such as video processing in TV sets, and games.

Embedded systems interface with the real world, where time is often a crucial factor, either to sample the environment or to react to it. Predictable timing can be essential for safety-critical systems, such as ABS brakes and avionics. The time factor is also of paramount importance for the "disappearing computer" [Lee2009], a.k.a. ambient intelligence. In this case the computer has to completely blend in with the physical world, and therefore must fully operate in "real time", or at least within a time window not noticeable by humans. Predictable and fixed latency is also of paramount importance in some cases: it is impossible for human to play an interactive game with a constantly varying latency. The lowest latency is also the key of success for high-frequency trading, where traders' datacenters are built near the stock exchange data center in order to further decrease latency.

For large-scale parallel applications predictable timing is essential for performance. If parallel tasks do not have the same execution time they can cause dramatic load imbalances. Even slight variability in latency can cause cascading changes in processor usage, which can result in dramatic power fluctuations in large systems. Controlling and understanding execution time is therefore essential for effective large-scale computing.

Guaranteeing fault tolerance under the condition of real-time correctness, and without over-provisioning resources is a serious technical challenge for which there are no clear-cut solutions at this moment.

Safety and security: Mission critical systems such as systems in aircrafts and satellites usually rely on simple, highly redundant architectures with predictable behavior in such a way that full deterministic behavior can be validated by certification authorities. Several factors challenge this approach. The increasing performance needs, the need for more complex processing as well as the need for higher integration (e.g., Integrated Modular Avionic (IMA)) make the use of older, more predictable, processors and techniques with high overhead increasingly costly. On the other hand, we can also foresee a proliferation and an increasing need for critical functionality in more and more aspects of day-to-day life. Functionality such as car control and healthcare monitoring are starting to become realities in our lives. Such a proliferation represents an opportunity to find more generic solutions for economically feasible safe and secure systems for the large public rather than only for the limited market of the avionic and space industry.

D. HiPEAC Research Areas in Architecture, Compilers, and Systems

The following section describes detailed research areas and topics critical to achieving the HiPEAC Core Computing Systems Challenges. The relationship between these research areas and the HiPEAC Core Computing Systems Challenges is summarized in the following table.

D.1. Parallelism and Programming Models

D.1.1. Locality Management

Communication is always expensive, in terms of energy, delay, and infrastructure cost. Thus, scalability and performance require optimizations to keep data local. Locality has been traditionally managed via either *shared memory or message passing.* Modern research, however, has moved beyond this dichotomy by describing memory as a *Shared Address Space*. Within a shared address space, if caches are coherent, they implement "shared memory". On the other hand, distributed memories within a shared address space communicate most efficiently using *remote DMA (RDMA),* which is an improvement over "message passing". The key distinction between these two schemes is whether communication is implicit (as with coherent caches) or explicit (as with RDMA).

Communication is *implicit* when the addresses supplied by the software do not identify physical data locations or (time of) movement. This has the advantage of simplifying the software, while disadvantages include: *(i)* hardware does not always have the information needed to make the best decisions; *(ii)* even in cases where software knows something more than hardware, software is unable to convey such knowledge to the hardware; and *(iii)* coherence protocols, directory operation, and communication granularity consume extra traffic and energy. Communication is *explicit* when software (the application, or compiler, or runtime system) is able to also indicate physical placement or transfers of data, besides specifying computation. This is an extra burden on the software, but it may lead to improved performance, reduced energy, or both, in cases when software is able to exploit its additional knowledge compared to hardware.

		Heterogeneous computing systems	Locality and communications management	Software for heterogeneous multi-cores	Cross-component/cross-layer optimization	Next -generation processing cores	Architectures for the Data Deluge	Reliable systems for Ubiquitous Computing
9.1. Parallelism and Programming Models								
9.1.1.	Locality Management	x	x	x	x	x	x	x
9.1.2.	Optimizations programmer hints, tuning	x	x	x	x			x
9.1.3.	Runtime Systems and Adaptivity	x	x	x	x		x	x
9.2. Architecture								
9.2.1.	Processors, Accelerators, Heterogeneity	x	x			x		x
9.2.2.	Memory Architectures	x	x		x	x	x	
9.2.3.	Interconnection Architectures	x	x		x	x	x	
9.2.4.	Reconfigurability	x	x		x	x		
9.3. Compilers								
9.3.1.	Automatic Parallelization		x	x		x		
9.3.2.	Adaptive Compilation			x			x	x
9.3.3.	Intelligent Optimization			x	x	x		x
9.4. Systems Software and Tools								
9.4.1.	Virtualization	x		x	x		x	x
9.4.2.	Input, Output, Storage, and Networking		x		x		x	
9.4.3.	Simulation and Design Automation Tools	x			x	x		
9.4.4.	Deterministic Performance Tools	x	x	x	x	x	x	x

Software for future many-core systems should bridge the dichotomy between efficiency and productivity, by providing sufficient abstractions for the productivity-oriented programmer to develop correct parallel programs with reasonable effort, and sufficient means for the efficiency-oriented programmer to optimize parallelism and data locality. Programming models based on the Partitioned Global Address Space (PGAS) abstraction can meet these criteria. A global address space simplifies the naming of data. At the same time, PGAS models make the physical location of data explicit to the efficiency-oriented programmer or systems software, who can control data placement, replication, and migration to avoid costly remote memory accesses.

While PGAS models appear to provide the right balance between efficiency and productivity, they still fall short of message-passing models (such as MPI) on architectures with no system-wide cache coherency. They also perform worse than shared-memory models (such as OpenMP, or Cilk) on architectures with system-wide cache coherence. The first performance gap arises because of the inability of current PGAS compilers and runtime systems to optimize explicit communication at the same level as an expert programmer. The second gap arises because of the inability of PGAS compilers to generate code that accesses data at bare-metal speed, when data is available to the local memory hierarchy of a processor. We believe that future research in compilers and runtime systems for parallel programming should address the above challenges.

PGAS models can leverage both implicit and explicit communication in a unified runtime environment, to manage effectively both regular and irregular data access patterns. A promising new direction for PGAS models is dynamic data-flow, which aids programmers both in the discovery of parallelism and the management of data locality. This model uses data access annotations to assist the compiler and runtime system in automatically analyzing data dependencies between regions of code, thus properly scheduling concurrent execution. At the same time, annotations associate data with computation, allowing the automation of data transfers (e.g. prefetch/presend, coalesce, adaptive path selection).

D.1.2. Optimizations, programmer hints, tuning

Compilers and runtime systems play a major role in optimizing code. Current optimizing compilers can extract thread-level or SIMD parallelism, or schedule loops to exploit locality. However, despite significant research efforts into auto-parallelization, quite modest progress has been made over the past thirty years. At the same time, aggressive speculative or predictive approaches to extract thread-level parallelism on the fly have shown that parallelism not extractable by static approaches can be exploited at run-time. Related to this parallelization issue, system dynamicity also needs intelligent runtime support for dynamic load balancing, power and thermal optimization of the system. The main concern in this field is therefore dealing with global optimization of the system performance, regardless of the computation model, while sustaining low overhead in low-level management of the hardware resources.

Most parallelization frameworks conform to sequential semantics when extracting parallelism. This model is in many cases too conservative. Programmer hints based on application domain knowledge can eliminate dependences that would otherwise disable parallelism from being extracted. Recent research advances, e.g. in commutativity analysis, have shown that significant amounts of parallelism can be unlocked by providing the compiler/runtime system with such programmer hints. Programmer hints, or compiler directives, have further been used to direct compilers to automatically create code for data movement and execution in heterogeneous systems. Such compiler directives are a promising medium-term solution to the difficulty in providing full automatic parallelization.

However, directives can be brittle, and cause unexpected behavior and dependencies with other portions of the code or libraries. This is particularly the case in the context of tuning for application performance. Further integration is required between the compiler and runtime system to avoid these issues.

D.1.3. Runtime Systems and Adaptivity

The runtime system has become a critical component of the software stack of parallel computer architectures. Since the introduction of multi-core processors, runtime systems have evolved from passive language libraries to sophisticated, dynamic optimization tools. Future runtime systems must form an extremely thin and painstakingly optimized layer of system software, which will enable parallel code to run correctly and efficiently on any parallel architecture. Runtime systems will require new methods to simplify the development and debugging of parallel code, while improving performance, scalability, and energy-efficiency. Given the diversity of parallel architectures, runtime systems will face the challenge of achieving portable performance on a wide range of systems, with multiple forms of parallelism, using both implicit and explicit communication. Runtime systems will also need to adapt to fluctuating and often unreliable hardware components. They will need to manage deep memory hierarchies, with highly non-uniform latency and they will need to sustain high performance under a tight energy budget.

Future runtime systems must address the issue of code portability and optimization across devices with different archi-

tectures, such as CPUs and GPUs. These devices require both different binaries and different optimizations. As the target device may not be known at compile time, or may change at runtime due to load balancing, the runtime must be able to efficiently compile and optimize code just in time (JIT). Effectively carrying out such optimization requires fast, accurate power and performance models for the underlying hardware. Without such models the runtime system has limited ability to make intelligent decisions, particularly with regards to the benefits of costly operations such as re-optimizing code for a different device.

Runtime systems must be aware of the state of the whole system to prevent oversubscription and conflict across applications. This requires either global coordination among different applications, potentially at the operating system level, or intelligent load analysis and back-off at the library level. Without such system awareness, two applications using intelligent runtime systems can both attempt to use the whole machine, thereby reducing performance due to oversubscription. The runtime system should be able to accurately control the performance of the core according to the needs, for power optimization by tuning the clock frequency and the voltage (DVFS).

Another challenge, more related to embedded systems, is to keep the expected timing properties on a highly dynamic software environment managed by the runtime systems. The runtime system should obey to meta-rules keeping the correctness of the non-functional properties of the system.

To fully leverage runtime systems, compilers and programming systems must be aware of the services they provide. Such integration has several benefits. From a productivity point of view, language integration makes the runtime system easier for developers to use. This results in simpler development and fewer programming errors. From a performance point of view, if the compiler and language are aware of the services provided by the runtime layer, they can take advantage of this to produce more efficient code. Additionally, an understanding of the runtime semantics at the language level will enable easier and better formal verification of an application's correctness.

D.2. Architecture

D.2.1. Processors, Accelerators, Heterogeneity

The hardware/software interface – known as the instruction set architecture (ISA) – has changed little due to the need to maintain backward compatibility. With the move from sequential computers to parallel multi-core architectures, this interface must be reevaluated. Since parallelism has become the dominating factor moving forward, it is important to revisit this interface with respect to what additional primitives are needed to assist the development of parallel software, and how they can be supported efficiently at the architecture level. A true challenge is to identify which primitives should be exposed to advanced programmers and compilers, and how to support them efficiently. Such primitives fall into categories such as thread management, synchronization support, and monitoring of parallelism, performance, and energy.

Traditionally the operating system has been responsible for the management of resources such as threads and memory. However, with processors containing multiple heterogeneous processor cores and complex memory hierarchies, these on-chip resources must be controlled at a much finer time-scale than current operating systems are capable of doing. As a result, there is a need to revisit the interface and division of responsibilities between the hardware architecture and operating system in terms of resource management. In addition to being essential for performance, this interface and division of responsibilities plays an essential role in providing the predictable hardware execution needed to guarantee safety-critical system performance.

D.2.2. Memory Architectures

Impact of memory latency: As processor chips have moved from frequency scaling to core replication, the latency gap between processor and memory has stopped increasing. In fact, since access time of DRAM keeps decreasing, latency may even decrease. On the other hand, as technology continues to shrink, the latency to traverse the on-chip interconnect and memory hierarchy is increasing, and contention for the shared off-chip bandwidth compounds the effect. In addition, core specialization (customization) will increase the computation speed relative to memory speed. These factors suggest that memory latency, as perceived by individual load/store instructions, will remain a problem in the years to come.

Therefore, a continued focus on reducing the average effective access latency to memory is warranted. This includes research into cache and memory locality management, including specific control over cache usage and data placement, and automated or software-guided data prefetching/presending, all of which aim at improving resource utilization and the effectiveness of on-chip memories.

Impact of memory bandwidth: For future chips with hundreds or thousands of cores, the on-chip memory subsystem, the network-on-chip (NoC), and the interface to the off-chip world are critical resources. Scaling these subsystems in an efficient manner to accommodate future increases in core count is a major challenge.

According to the ITRS, off-chip bandwidth is expected to increase linearly rather than exponentially. As a result, new

methods are needed to better exploit the limited on-chip storage, thereby reducing the frequency of accessing off-chip data. Such locality optimizations are also crucial because the energy needed to move data off-chip is much higher than to move it inside the chip. A related issue is how to allocate the available bandwidth across the cores for fairness, energy efficiency, and predictability.
Locality management is an important approach to reducing off-chip bandwidth, and it needs to be pursued at all levels: application, programming model, compilation, runtime system (libraries), and hardware architecture. On-chip memories, including caches and local memories, are crucial in this respect. We have seen a diminishing return on investments in the real estate devoted to caches, so clearly cache hierarchies are in need of innovation to make better use of their resources.

Cache management will continue to be an important topic of research, as will be the emerging, more general, topic of local (on-chip) memory management. Also, when trading cores versus local memories and caches in the future, the designer will have to seriously consider balancing not only silicon area but also power envelope costs. One potentially important approach being studied to mitigate the limited off-chip bandwidth is the prospect of 3D stacking, which enables tight integration of substantially more memory resources to reduce off-chip memory bandwidth.

An interesting new perspective is to look at future chips as a sea of (volatile and non-volatile) memory blocks, interspersed with processing power (cores). In this model, application data is stored in the memories, which provide parallel access. Whenever we want to operate on the data, we find a *nearby idle processor,* power it up, and perform the operation local to the data, rather than moving the data to the processor.

Scalable shared address space, with implicit and explicit communication: A great deal of attention was devoted to scalable cache coherence protocols in the beginning of the 90's, and enabled industrial offerings of shared memory multiprocessors with a processor count of several hundred, e.g., SGI Origin 2000. More recently, the latency/bandwidth tradeoff between broadcast-based (snooping) and point-to-point based (directory) cache coherency protocols has been studied in detail. It is now well understood how to scale cache coherence, in a fairly efficient way, to systems with a hundred processors, although scaling it to thousands of processors has not been demonstrated, and is truly challenging.

Now that we are approaching systems with hundreds of cores on a chip, technological parameters and constraints will be quite different. For example, cache-to-cache miss latencies are relatively shorter, and the bandwidth available on-chip is much larger than for the "off-chip" systems of the 90s. On the other hand, design decisions are severely constrained by power consumption. All these differences make it important to revisit the design of scalable cache coherence protocols for the multi-cores in this new context. An important research focus is on scalable on-chip cache coherence, which has to consider new technology parameters as well as the important constraint on power consumption.

At the other end of the spectrum, supercomputers have always been built without (large scale) cache coherence, and explicit communication (message passing) is used for their operation. There are also commercial many-core chips that have adopted the explicit communication model, e.g. the IBM Cell and the Intel SCC (single-chip cloud), in order to decrease design complexity, increase scalability, and allow improved performance in cases where software is able to explicitly manage coherence. However, none of these approaches have been popular with developers due to the extreme difficulty of manually analysing, managing, and optimizing data movement.

As research progresses, the spectrum of techniques for improved locality management will become richer. On one hand, more effective heuristics will be discovered for *implicit communication*, through cache coherence, to better adapt to the circumstances, thus providing higher performance, at lower cost, for a wider set of cases. On the other hand, runtime libraries will become increasingly able to efficiently manage locality, using software algorithms that are more sophisticated than their hardware counterparts, while relying on hardware primitives that give *explicit control* of the communication to the software. This is a key task for *efficiency programmers*, who focus on optimizing critical pieces of code; productivity programmers, then, use these pieces in their code, with significant gains in performance. In some cases, the application is so important that it is justified to have efficiency programmers take the effort to explicitly manage locality. In the majority of cases where this is not the case, however, we must ensure that some combination of the compiler, runtime system, and hardware ensure that implicit communication is available.

Research is needed in all of the above models, as we are moving into architectures that unify both implicit and *explicit* communication – each for its own distinct purposes – under the common umbrella of *Shared Address Space*.

D.2.3. Interconnection Architectures

Interconnection networks are essential components of computing systems, enabling the communication (a) of processors to their memories, (b) among the levels of the memory hierarchy, (c) processors or compute engines to each other,

both inside each chip and across chips, (d) communication to storage and I/O devices, and (e) intra- and inter-system communication. Although the general principles of switch, router, and network architecture are the same across all levels, technologies and constraints vary widely, resulting in radical differences among networks-on-chip (NoC), chip-to-chip interconnects, and system, local, or wide area networks.

Interconnection architectures undergo rapid changes, as a result of (a) technology evolution, especially CMOS scaling, chip packaging including 3D-stacking, and new optics (photonics on silicon); (b) advances in the theory and practice of minimizing latency, energy consumption, and area cost; (c) requirements to support new functions, including reconfigurability, fault tolerance/dependability in the presence of unreliable components, and predictability; and (d) the requirements of new applications and programming methods.

Improving interconnection networks requires large and continuous efforts on multiple fronts.

- For on-chip networks: router, network-interface, and memory-controller microarchitecture; topologies beyond basic meshes and trees; routing algorithms, including adaptive, multipath, fault-tolerant, application-specific, predictable latency, reconfiguration, and circuits, including multiple clocking/power-off islands, differential signaling, integrated channel buffers, and split and variable size data paths; buffering and flow control for QoS, virtualization, and security; traffic, power, and error monitoring and debugging support; design and simulation tools and models, including for full-system simulation and application-driven traffic modeling.
- For DRAM and persistent storage communication: increased throughput and improved topologies; packaging technologies including 3D stacking; packet-oriented protocols.
- For multi-chip systems: proper combination of electronics and optics, depending on technology evolution (this may eventually also affect on-chip networks); dynamic power management; protocols, topologies, and routing; switch, buffering, QoS, and related architectures.

D.2.4. Reconfigurability

Reconfigurability has the potential to combine software-like flexibility with the high-performance ability of hardware. Reconfigurable devices enable a massive number of programmable computational nodes that execute in parallel, providing substantial performance benefits over conventional general-purpose machines. Their power/performance ratio can often be considerably better than a general purpose processor as they adapt to the specific needs of an application supporting customization of the datapath and control-flow of hardware. In particular, this adaptability can provide for far lower and more predictable latency for time-critical tasks. On-demand reconfiguration can be also exploited to isolate and correct defective blocks offering an excellent solution for fault-tolerance.

For these reasons, reconfigurable hardware is becoming popular in embedded systems and application-specific designs, while it can also provide an attractive solution for high-performance computing. Commercial solutions for integrating commodity reconfigurable hardware have appeared for the HPC domain from companies such as Maxeler, Convey, and Cray.

However, in order to deploy this technology more successfully in more application domains, several challenging issues must be addressed. The most important are improved ease of programming and reduced overhead for configurable switching and logic. Additional challenges include better leveraging of runtime reconfiguration, customizable power-efficiency, adaptive memory hierarchy, and improved off-chip bandwidth compared to GPUs and CPUs. In order to address these challenges and to ease the usage of reconfigurable technology, a new generation of efficient architectures, tools, methods, and runtime support are fundamental. CGRA is an interesting direction by combining the efficiency of more coarse grain optimization of compute nodes with the reconfigurability of FPGAs. In case of a reconfigurable system the above is even more challenging due to the hardware polymorphism, hence holistic hardware/software approaches are essential.

Furthermore, reconfigurable computing platforms have the potential to play an important role in supporting fault tolerant computing. Their inherent flexibility can be deployed to correct, isolate, and/or replace faulty parts of a system. Efficient techniques can be developed and when combined with runtime support may result in a fault tolerant system that can recover from transient faults or cope with permanent faults preserving critical functionalities of the system.

D.3. Compilers

D.3.1. Automatic Parallelization

As multi-core processors become more and more ubiquitous in all computing domains, programmers will design and write parallel applications to take advantage of their compute performance. However, although multi-core processors have been mainstream for several years, current software development still lags behind, with sequential applications still the dominant program design. The underlying reason for this is that parallel programming is far more difficult than sequential programming. In particular, the cost and time required to parallelizing existing sequential applications is nearly prohibitive for all but the most performance-critical domains.

An alternative to manually parallelizing applications is automatic parallelization, whereby the compiler identifies the regions of code that can be run in parallel and automatically splits the program to enable their concurrent execution. In recent years there have been significant advances in automatic parallelization research. In particular, polyhedral models of complex loop nests and the associated techniques needed to choose the correct optimization have advanced significantly. However, performance of these schemes is still disappointing, largely due to the imprecision of the input language (often C) and the inherent irregularities of many computations. As a result, no automatic techniques today can truly take advantage of the execution power of modern multi-core machines across a wide range of applications. Advancing the state-of-the-art in this area would have significant impact across all computing domains.

D.3.2. Adaptive Compilation

In the past, applications were compiled with the goal of maximizing performance only, with the target end system known well in advance.

Nowadays programs must run under varying conditions that cannot be known until execution time, such as power constraints and resource contention from other applications, and the end system may span a wide range of systems, including mobile, desktop, and the cloud. In this context, adaptive compilation becomes increasingly important, as applications can be tuned once the final system is known and then re-optimized as runtime conditions change.

The most successful examples of adaptive tool-chains are Just-In-Time (JIT) compilers such as the Java Virtual Machine and OpenCL, which compile each piece of code immediately before it is required. JITs allow code to be written once and run anywhere and will become increasingly important to allow dynamic optimization and adaptation to the current system conditions.

The widespread adoption of JITs presents two significant concerns. First, for the massive deployment of these technologies in embedded systems, efficient hardware and software runtime support for JITs must be defined. Second, while JITs provide code portability across devices, experience with OpenCL has clearly indicated that they do not provide performance portability across widely varying architectures. More research is needed to develop more intelligent optimizations that provide true portability.

D.3.3. Intelligent Optimization

Modern compilers contain a wide variety of sophisticated optimizations that can be used to create binaries that are highly tuned for a specific platform. However, the majority of optimization passes has hard-coded parameters set by the compiler writer and are executed in a fixed order that may not produce the most optimal code for each application or platform. Due to the sheer number of optimizations available and the range of parameters that they could take, it becomes impossible to identify the best sequence by hand.

An automatic method is required for learning the best optimizations to run, their order and their parameters for each program on each target system. Recent work has shown how this intelligence can be included in the compiler through the use of machine learning. However there is significant work still to be done to harness the full power of each optimization for any system [COLE,CTUNING]. The potential for intelligent optimization is greatly increased when combined with runtime systems and JIT compile code. As such code is compiled on-demand, the compiler system can choose optimization for the final target platform, and the runtime system can analyze the impact of those optimizations, potentially deciding to re-compile if further optimization is warranted.

D.4. Systems Software and Tools

D.4.1. Virtualization

Virtualization is a technique that creates a software abstraction for a hardware device. The same software abstraction can be supported on different types of hardware, and it allows the same application image to run on a variety of hardware platforms. The standardized application state facilitates the reification of the application state, and thus also the migration of applications between machines.

Virtualization is a state of the art technique in data centers where it is used to consolidate workloads on fewer servers during off-peak hours and leads to significant operational cost savings. At the same time, the isolation provided by virtual machines also leads to improved security and fault tolerance at the system level, and reduces the IT management costs. It also finds its way to desktops where it is used to reduce system management costs, and to run multiple software platforms on the same physical hardware. Traditional virtualization techniques are a mature technology, especially in cases where it is supported by dedicated hardware.

However, the proliferation of heterogeneous hardware computing platforms and the need for predictable performance for timing-critical systems is widening the gap between the virtual machine and the physical hardware. This gap creates several challenges for virtualization technology:

- Portable performance: How to efficiently map (platform-independent) application images on a heterogeneous hardware platform with a variety of accelerators.

- Accelerators: How to efficiently virtualize accelerators.
- Scheduling: The development of performance models for virtualized workloads, needed for scheduling and mapping.
- Trustworthiness: How to guarantee full isolation between different applications that are consolidated on a single hardware platform. How prove that a hypervisor is secure, how to certify it.
- Predictability: How to provide predictability guarantees for applications on virtualized hardware.

D.4.2. Input, Output, Storage, and Networking

The "data deluge" from scientific applications, sensors, and online transactions is of paramount importance for emerging and future applications and infrastructures, and in many cases constitutes the main application bottleneck. For storage I/O, new technologies in persistent memories, such as flash or byte addressable persistent memories, present an opportunity to move persistence closer to the CPU. This will have a profound impact on system-level architecture as well as on the whole software stack, both in terms of performance and reliability.

Similarly, network I/O in large-scale systems will require new techniques, abstractions, and architectures to achieve the efficiency required for processing large amounts of data. TCP/IP- and Ethernet-type network I/O will require significant improvements to keep up with the increasing number of cores. To ensure that the network will not be a bottleneck for future applications, we must ensure high communication protocol efficiency at the edge of the network, proper dimensioning of the network, and dynamic network adaptation.

D.4.3. Simulation and Design Automation Tools

Raising abstraction levels and accelerating synthesis and verification can enable designers to explore larger design spaces more efficiently and rapidly. This trend has been the key factor in the evolution of design and optimization tools. Although many design techniques have been developed over the last decades for the current generation of electronic devices, the soaring complexity of electronic systems will soon require new evolutions in exploration, design and verification processes.

To manage the complexity of systems and to reduce the design cycle time, new and more efficient methodologies, leveraging current ESL approaches, are needed to create the future generation of electronic devices. They will have to break the trend to compromise on the evaluation of various design implementation options. Future generations of EDA/CAD tools will have to automatically generate optimized and functionally correct implementations of electronic system for both the hardware and software parts. Seamless design flows based on virtual prototyping (simulation, digital mockup, …), design space exploration (multi-objective optimization), system synthesis (high-level synthesis, software synthesis, …) and (semi-)formal verification are required to generate complete systems.

These tools could rely heavily on the ability to simulate system behavior efficiently and accurately, at both the software and hardware level. As the most detailed device-level or cycle-accurate simulations are far too slow for productive use on large-scale systems, simulation technology must explore means to parameterize device and system behavior to allow efficient design space exploration. Fast statistical models, interval simulation, and hierarchical simulation are all promising methods for trading off detail and performance while retaining accuracy. However, for these techniques to be trustworthy, they must be carefully evaluated against representative, and diverse, implementations of real systems.

D.4.4. Deterministic Performance Tools

Predictable timing is essential for safety-critical systems as well as performance for large-scale parallel systems. We therefore need to promote the notion of time as a first-class notion across the whole computing stack: programming languages should allow the programmer to express timing constraints, and compilers and virtual machines should enforce these constraints using timing information produced by the hardware. Further, measured statistical timing information should be tied back to these constraints to assist with performance and correctness debugging.

This can be achieved by using well-defined time-aware models of computation. Such models of computation have been proposed for the embedded systems (Ptolemy, synchronous languages) but should be revisited taking into account future heterogeneous parallel architectures and should be extended to the general-purpose computing domain where timing will be important to energy-efficiency. Indeed, a processor can be slowed down if the result of the computation is not needed right away for external reasons (timing constraints) or internal reasons (synchronization with other concurrent computations). Challenges are:

- Expressiveness of the timing specification constructs
- Determinism of the computation models
- Efficiency of the runtime monitoring and scheduling
- Trustworthiness of the runtime system
- Integration with commercial tools and hardware to provide predictable execution

Glossary

Adaptive Compilation
Adaptive compilation is a compilation process whereby the generated code is adapted to changes in the underlying hardware. These can both be short-term changes, such as cache misses, or long term changes, such as the increasing number of available cores. Both online and offline techniques are used.

ASIC
Application-Specific Integrated Circuits are integrated circuits designed for a particular purpose, as opposed for general use in many different situations.

ASIP
An *Application-Specific Instruction-set Processor* is part of a system-on-a-chip. Its instruction set is designed to be optimal for the applications targeted by said system. It is programmable and hence more flexible than an ASIC, but still offers a better performance/energy trade-off than generic processors due to its specificity.

Binary Translation
Binary Translation is transformation process during which machine code is rewritten, and can either keep or change the ISA. This process can either happen offline (statically) or online (dynamically, at run time).

Byte code
Byte code refers to an intermediate format in which executable code is stored. This intermediate format can subsequently be read and be transformed in combination with other byte code. It can also be interpreted, or statically or dynamically translated into machine code.

CAGR
Compound annual growth rate is a business and investing specific term for the smoothed annualized gain of an investment over a given time period.

CGRA
A *Coarse Grain Reconfigurable Array* is essentially an array of processing elements (PEs), connected by a 2-D network.

Cloud computing
Cloud computing is a paradigm whereby computing power is abstracted as a virtual service over a network. Executed tasks are transparently distributed.

Compiler
A *compiler* is a computer program that transforms higher level program representation into a lower level one.

Composability
Composability refers to the ability to maintain the properties of independently designed components (response times, maximum throughput, …) when they are combined in a single system.

Computing Systems
Computing Systems are the common denominator for embedded computing, general purpose computing, high performance computing, and also refers to hardware and software.

Customization
Modifying the operation or properties of a design to the actual needs is called *customization*.

Disappearing computer
As an increasing number of common appliances include more and more processing power, the use of traditional computers may well diminish over time. In a matter of speaking, the computer is being absorbed into all

DSE
The number of tunable parameters in chip design continuously grows. In *Design Space Exploration*, the effects of changing these parameters are studied. As the interactions between the different parameters have become too complex for humans to properly predict the effects of changing them, automatic DSE is becoming increasingly important.

DSP
Digital Signal Processors are processor cores optimized for signal processing and not general purpose computation. They often have special purpose functions for accelerating common communications and image processing standards.

DVFS
Dynamic Voltage Frequency Scaling refers to increasing or decreasing the frequency and voltage applied to chips. When the voltage is lowered, it takes longer for the circuits to switch, so the frequency must be reduced, but power consumption is lower. The opposite effect can be achieved by increasing the voltage.

FGPA
Field Programmable Gate Arrays are the dominant form of commercially available reconfigurable logic devices and are used for low-volume and latency-sensitive applications.

GPU
A *Graphics Processing Unit* refers to the processing units on video cards. In recent years, these have evolved into massively parallel execution engines for floating point vector operations, reaching performance peaks of several gigaflops.

HiPEAC
The European Network of Excellence on *High Performance and Embedded Architecture and Compilation* coordinates research, facilitates collaboration and networking, and stimulates commercialization in the areas of computer hardware and software research.

ICT
Information & Communication Technology is a generic term used to refer to all areas of technology related to computing and telecommunications.

ISA
An *Instruction Set Architecture* is the definition of the machine instructions that can be executed by a particular family of processors.

JIT
Just-In-Time compilation is the method of compiling code from source or an intermediate representation at the time when it will execute. This allows for improved portability by generating the correct binary at execution time, when the final target platform is known. JIT compilation has been heavily leveraged in Java, Microsoft's C#, and OpenCL.

Mobile convergence
As portable devices become more able and better performing, they are able to take on more and more tasks that used to be performed by desktop computers or even servers. At the same time, the energy wall is requiring traditional computers to scale down. Where these two trends meet, we talk about *mobile convergence*.

Moore's law
Moore's law was defined by Intel co-founder Gordon Moore. It describes the trend that the number of transistors on a chip doubles every 18 months. As other trends such as performance and storage capacity are strongly related to this evolution, it is often also used in those contexts.

Multi-core
When multiple processor cores are placed onto a single chip, we talk about a *multi-core* processor. Each of those processor cores can either be identical (*homogeneous* multi-core), or some can differ from others (*heterogeneous* multi-core).

NRE
Non-Recurring Engineering costs refer to one-time costs incurred for the design of a new chip, computer program or other creation, as opposed to marginal costs that are incurred per produced unit.

Out-of-order processor
An *out-of-order processor* does not execute all machine instructions in the same order as they appear in the program, in order to avoid stalls caused by having to wait for a particular instruction to finish. It can only do so if it can guarantee that the behavior of the program will not change, which requires costly run-time dependency analyses.

PGAS
Partitioned Global Address Space is a parallel programming model. It assumes a global memory address space that is logically partitioned and a portion of it is local to each processor. The novelty of PGAS is that the portions of the shared memory space may have an affinity for a particular thread, thereby exploiting locality of reference.

Predictability
Real-time applications often have very stringent constraints regarding how long particular operation may take, or the maximum response time between an event and handling it. Being able to guarantee that the execution will fulfill these requirements under all circumstances, requires a large amount of *predictability* insofar the execution is concerned.

Programming model
A *programming model* is a collection of technologies and semantic rules that enable expressing algorithms in an efficient way. Often, such programming models are geared towards a particular application domain, such as parallel programming, real-time systems, or image processing.

Reconfigurable computing
Reconfigurable Computing relies on hardware that supports arbitrary functionality on demand. Such static or dynamic hardware customized is used to meet various system and application requirements. All the necessary design tools, run-time system extensions, methods and programming models are also considered.

SoC
A *System on Chip* refers to integrating all components required for the operation of an entire system, such as processors, memory, and radio, on a single chip.

Soft Errors
A *soft error* is a temporary wrong result, often caused by cosmic rays or temperature effects, not by a permanent failure of the circuit (which is called a hard error). With increasing integration the likehood of soft errors will increase.

STDP
Spike-Timing-Dependent Plasticity is a biological process that adjusts the strength of connections between neurons in the brain. The process adjusts the connection strengths based on the relative timing of a particular neuron's output and input action potentials (or spikes).

System in Package (SiP)
A *System in Package* consists of multiple, vertically stacked dies that are combined to provide the functionality of an entire system. The difference with a System on Chip is that here the third dimension is also used.

Telepresence
Telepresence is the concept of appearing to be in another place than where they are physically located. Possible aspects include giving the actor the impression that they are elsewhere, giving people in another location the impression that the actor is with them, and giving the actor the ability affect a remote environment.

Virtualization
Virtualization encompasses a large number of technologies, all geared at abstracting some underlying components. This can range from a software layer to an ISA to I/O devices to hardware defects.

References

[Aitken2011] M. Aitken, K. Flautner, and J. Goodacre, "High-Performance Multiprocessor System on Chip: Trends in Chip Architecture for the Mass Market," in Multiprocessor System-on-Chip, Hardware Design and Tool Integration, New York, NY: Springer New York, 2011, pp. 223-239.

[Alibart2010] F. Alibart at al. "An Organic Nanoparticle Transistor Behaving as a Biological Spiking Synapse," Adv Functional Materials 20.2, 2010.

[ARWU2011] http://www.arwu.org/, retrieved on 1/9/2011.

[Avizienis2004] A. Avizienis, J.-C. Laprie, B. Randell, C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," IEEE Transactions on Dependable and Secure Computing , vol.1, no.1, pp. 11- 33, Jan.-March 2004.

[Borkar2004] S.Y. Borkar, "Microarchitecture and Design Challenges for Gigascale Integration," in 37th Annual International Symposium on Microarchitecture (MICRO-37 2004). p. 3, IEEE Computer Society, 2004.

[Borkar2005] S.Y. Borkar, "Designing reliable systems from unreliable components: The challenges of transistor variability and degradation," IEEE Micro, 25(6):10–16, 2005.

[CM-1] D. Hillis, "The Connection Machine," MIT Press Series in Artificial Intelligence, 1985.

[COLE] K. Hoste and L. Eeckhout. "Cole: compiler optimization level exploration," in Proceedings of the 6th annual IEEE/ACM international symposium on Code generation and optimization (CGO '08). ACM, pp. 165-174, 2008.

[CTUNING] G. Fursin and O. Temam. "Collective Optimization: A Practical Collaborative Approach," in ACM Transactions on Architecture and Code Optimization (TACO), Volume 7, Number 4, pages 20-49, 2010.

[DarkSilicon] http://www.eetimes.com/electronics-news/4085396/ARM-CTO-power-surge-could-create-dark-silicon- or http://www.cadence.com/Community/blogs/ii/archive/2010/03/31/arm-keynote-will-dark-silicon-derail-the-mobile-internet.aspx, retrieved on 1/9/2011.

[DISTIMO] https://monitor.distimo.com/, retrieved on 1/9/2011.

[DNA] C. Dwyer and A. Lebeck, "Introduction to DNA Self-Assembled Computer Design", Artech House, Norwood, 2007.

[Dubey2005] P. Dubey, "Recognition, mining and synthesis moves computers to the era of tera," in Technology@Intel Magazine, 9(2):1-10, 2005.

[E10] "The data deluge: businesses, governments and society are only starting to tap its vast potential" (special report on "Data, data everywhere"). The Economist. Feb 2010. http://www.economist.com/node/15579717 and http://www.economist.com/node/15557443.

[EMC20010] EMC-sponsored IDC study: "The Digital Universe Decade – Are you Ready?" May

2010. http://www.emc.com/about/news/press/2010/20100504-01.htm http://idc-docserv.com/925, retrieved on 1/9/2011.

[Esmaeilzadeh] H. Esmaeilzadeh, E. Blem, R. St. Amant, K. Sankaralingam, and D. Burger, "Dark Silicon and the End of Multicore Scaling," in Proceedings of the 38th International Symposium on Computer Architecture, San José, California, pp. 365-376, 2011.

[Fuller2011] S.H. Fuller, L.I. Millett, "The Future of Computing Performance: Game over or Next Level," Committee on Sustaining Growth in Computing Performance, National Research Council, National Academies Press, Sept, 2011.

[Gartner 2007] "Gartner Estimates ICT Industry Accounts for 2 Percent of Global CO2 Emissions," http://www.gartner.com/it/page.jsp?id=503867, retrieved on 1/9/2011.

[Geim2007] A.K. Geim, K.S. Novoselov, "The rise of grapheme," in Nature materials. 6(3):183-91, 2007.

[Google2009] http://googleblog.blogspot.com/2009/01/powering-google-search.html retrieved on 1/9/2011.

[HT03] A. Hey and A. Trefethen. "The Data Deluge: An e-Science Perspective," in F. Berman, G. Fox and A. Hey, Eds. Grid Computing - Making the Global Infrastructure a Reality, pp. 809-824. Wiley, 2003

[HTT09] T. Hey, S.Tansley, K. Tolle (eds.) "The Fourth Paradigm: data-intensive scientific discovery," Microsoft Research book. 2009. http://research.microsoft.com/en-us/collaboration/fourthparadigm/

[ICTWORK] "ICT Work Programme 2011-12" http://cordis.europa.eu/fp7/ict/docs/3_2012_wp_cooperation_update_2011_wp_ict_en.pdf retrieved on 1/9/2011.

[Intel] http://www.intel.com/content/www/us/en/silicon-innovations/moores-law-embedded-technology.html retrieved on 1/9/2011.

[ISTAG] "Shaping Europe's Future through ICT", ISTAG, March 2006.

[ITRS] "The International Technology Roadmap on Semiconductors 2009 Edition," Dec, 2009.

[KET] http://ec.europa.eu/enterprise/sectors/ict/key_technologies/index_en.htm retrieved on 1/9/2011.

[Kogge2008] P.M. Kogge (ed.), "ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems," University of Notre Dame, CSE Dept. Tech. Report TR-2008-13, Sept. 28, 2008.

[Kuzum2011] D. Kuzum, R. Jeyasingh, B. Lee, P. Wong, "Nanoelectronic Programmable Synapses Based on Phase Change Materials for Brain-Inspired Computing," in Nano Letters, June 2011.

[Lee2009] E. Lee, "Computing needs time," in Communications of the ACM, vol. 52, pp. 70–79, May 2009.

[Lightwave2010] "NYSE Euronext and 100G: The drive to zero latency" Lightwave, January 1, 2010, http://www.lightwaveonline.com/about-us/lightwave-current-issue/NYSE-Euronext-and-100G-The-drive-to-zero-latency.html

[P11] P. Ranganathan, "From Microprocessors to Nanostores: Rethinking Data-Centric Systems," Computer, vol. 44, no. 1, pp. 39-48, Jan. 2011. http://www.hpl.hp.com/news/2011_IEEEComputer_nanostores.pdf (also covered in NY Times "Remapping Computer Circuitry to Avert Impending Bottlenecks" 3/2011 http://www.nytimes.com/2011/03/01/science/01compute.html)

[Palem2005] 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, pp. 2-4, 2004.

[Pfister2007] G. Pfister, "Panel Position: Is the Multi-Core Roadmap going to Live Up to its Promises?", IDPDS, 2007.

[Pistol2010] C. Pistol, W. Chongchitmate, C. Dwyer, A. Lebeck, "Architectural Implications of Nanoscale-Integrated Sensing and Computing," Micro, IEEE , vol. 30, no. 1, pp. 110-120, Jan.-Feb. 2010.

[Rothemund2006] P. Rothemund, "Folding DNA to create nanoscale shapes and patterns," Supplementary Notes 1, Nature 440:297-302, 2006.

[Snider2008] G. Snider "Spike-timing-dependent learning in memristive nanodevices," in Proceedings of IEEE NANOARCH 2008, pp. 85-92, 2008.

[Spainhover1999] L. Spainhower, and T. Gregg, "IBM S/390 Parallel Enterprise Server G5 fault tolerance: A historical perspective," in Proc. of IBM Journal of Research and Development, pp. 863-874, 1999.

[Strukov2008] D. Strukov, G. Snider, D. Stewart, R.S. Williams. "The Missing Memristor found," Nature 453, pp. 80-83, 2008.

[SyNAPSE] https://www.ibm.com/smarterplanet/us/en/business_analytics/article/cognitive_computing.html retrieved on 1/9/2011.

[TAAB2008] "The Value of a Millisecond: Finding the Optimal Speed of a trading infrastructure" TAAB Group Report, 2008. http://www.tabbgroup.com/PublicationDetail.aspx?PublicationID=346 retrieved on 1/9/2011.

[WiGig] http://www.wigig.org/ retrieved on 1/9/2011.

[Whener2008] M. Wehner, L. Oliker and J. Shalf, "Towards Ultra-High Resolution Models of Climate and Weather" in International Journal of High Performance Computing Applications 22:149-165, 2008. or http://www.lbl.gov/Science-Articles/Archive/NE-climate-predictions.html

Acknowledgements

This document was written thanks to the valuable inputs from the HiPEAC members, the teachers of the ACACES summer school 2010 and 2011. The editorial board, composed of Marc Duranton (CEA), David Black-Schaffer (Uppsala University), Sami Yehia (Thalès) and Koen de Bosschere (Ghent University), would like to particularly thank:

Angelos Bilas (FORTH), Pierre Boulet (INRIA), Albert Cohen (INRIA), Philippe Coussy (CNRS), Raphaël David (CEA), Bjorn De Sutter (Ghent University), Pedro Diniz (inesc-id), Babak Falsafi (EPFL), Paolo Faraboschi (HP), Christian Gamrat (CEA), Georgi Gaydadjiev (TU Delft), Timothy M Jones (University of Cambridge), Manolis Katevenis (FORTH), Stefanos Kaxiras (Uppsala University), Jonas Maebe (Ghent University), Nacho Navarro (UPC), Dimitris Nikolopoulos (FORTH), Alex Ramirez (BSC), Per Stenström (Chalmers), Dirk Stroobandt (Ghent University), Olivier Temam (INRIA), Sid Touati (INRIA), Mateo Valero (BSC), Ayal Zaks (IBM).

The editorial board is also endebted to Dr. Panos Tsarchopoulos, project officer of the HiPEAC network of Excellence.

Computing Systems: Research Challenges Ahead
The HiPEAC Vision 2011/2012

HiPEAC COMPILATION ARCHITECTURE

Computing systems have had a tremendous impact on everyday life over the past decades in all domains. Historically, computing performance has been fuelled by "Moore's law", which drove the semiconductor industry for decades. However, a major paradigm shift is now taking place. "Moore's law", while keeping pace in terms of transistor density, will only enable a minor increase of the frequency and decrease of the power dissipation per transistor. As a result, even if it will still be feasible to pack more devices on a chip, it will not be possible to use them all simultaneously. New technology nodes are compounding this problem by increasing leakage power and device variability, and decreasing reliability.

The need to provide improved energy efficiency and build reliable systems from unreliable and highly variable components leads to new research directions at all levels. HiPEAC has identified seven specific research objectives:

Efficiency (with a focus on energy efficiency)

1) ***Heterogeneous computing systems:*** how can we design computer systems to maximize power efficiency and performance?
2) ***Locality and communications management:*** how do we intelligently minimize or control the movement of data to maximize power efficiency and performance?

System Complexity

3) ***Cost-effective software for heterogeneous multi-cores:*** how do we build tools and systems to enable developers to efficiently write software for future heterogeneous and parallel systems?
4) ***Cross-component/cross-layer optimization for design integration:*** how do we take advantage of the trend towards component-based design without losing the benefits of cross-component optimization?
5) ***Next-generation processor cores:*** how do we design processor cores for energy-efficiency, reliability, and predictability?

Dependability and applications (with a focus on their non-functional requirements)

6) ***Architectures for the Data Deluge:*** how can we tackle the growing gap between the growth of data and processing power?
7) ***Reliable systems for Ubiquitous Computing:*** how do we guarantee safety, predictability, availability, and privacy for ubiquitous systems?

Furthermore, it will be necessary to investigate research directions breaking with the line of classical Von Neumann systems. Fuelled by new technologies such as dense non-volatile memories, optical interconnects, and 3D stacking, new computing paradigms will be necessary to perform both old and new tasks at high efficiency levels while decreasing the impact of the constraints of the new technology nodes.

roadmap@hipeac.net
http://www.HiPEAC.net/roadmap