

## Chip-level integration: the new frontier for microprocessor architecture

Jaime H. Moreno  
IBM Thomas J. Watson Research Center  
Yorktown Heights, NY

Workshop on Parallelism in Algorithms and Architectures  
University of Maryland, College Park

May 2006

© 2006 IBM Corporation

### Abstract

- This talk reviews the reasoning behind the need for evolution in chip-level integration for microprocessors, and the challenges found in pursuing such objectives. It summarizes the need for identifying emerging tradeoffs, beyond those found in straight-forward multicore chips, as well as the need to determine how these tradeoffs can be investigated, and how proposed innovations can be incorporated through chip-level integration.

The talk focuses on describing challenges as opposed to proposing specific solutions, and illustrates those challenges with some examples; in doing so, the talk focuses on the description of opportunities in this emerging area, which appears ready for new research and development.

## Summary

- **Environment:** Serious challenges to high-end microprocessors due to limits in power consumption and power density, and decrease in silicon technology improvements
- **Assertion:** Opportunity for chip-level integration (“chip-level architecture”)
  - Improve computing capabilities through integration of multiple cores, heterogeneous cores, specialized functions, off-load engines, accelerators, memory hierarchy, interconnect fabric, etc.
    - Beyond packing together individual processor cores
  - Integration across all layers, from implementation level through microarchitecture up to system software.
  - Leverage experience from other areas (embedded systems in particular).
- **Will describe:**
  - Reasoning behind the need for migration towards chip-level integration for microprocessors, exploiting non-conventional directions, and challenges found in pursuing such objectives.
  - Need for identifying emerging tradeoffs, need to determine how such tradeoffs can be investigated, and how innovations can be incorporated through chip-level integration.
- **Scope:** Challenges instead of solutions, with some examples
  - Intention is description of field that is ready for new research and development, and for leveraging lessons learned elsewhere (SoCs).

## Outline

- Semiconductor technology background and motivation
- Opportunities for innovation through chip-level integration
- Concluding remarks

## Power consumption

- Components of power consumption
  - Active power
  - Passive power: transistor sub-threshold leakage
  - Gate leakage
- Passive power consumption continues to explode
  - Passive power has become comparable to active power
- Gate leakage is increasing rapidly
  - Driven by decreasing physical thickness of oxide

## Power consumption limits high-end microprocessors

- Technology improvements continue with respect to transistor density, although at a reduced pace
- No longer able to simultaneously utilize all transistors in a chip due to power limitations
- Need to focus on “efficiency”
- Workload demands are highly variable – no single design is optimal for all cases
- Need to accommodate diverse workloads while managing power constraints

## 3D integration

- Transition from bipolar to CMOS led to large reduction in power and increase in density but at the cost of 3x in transistor speed
- CMOS has reached power characteristics comparable to those of bipolar technology
- Low-voltage operation in CMOS can achieve reduction in power at reduction in performance (frequency) comparable to bipolar to CMOS transition
  - Density improvement can be achieved through three-dimensional silicon integration
- Advances in Si through-via technology and wafer bonding providing much higher I/O densities and bandwidth
  - Changes tradeoffs and potentially the conception of systems and software
  - What would we do if data cache transfers of KB/cycle was possible?

## eDRAM – another integration level

- Processor chips are currently cache starved due to area constraints
- Multi-core, multi-threaded and multi-image (virtualization) chips are stressing memory bandwidth and on-chip memory capacity
- eDRAM benefits over SRAM: density (4X), lower power, better SER characteristics
- eDRAM array access slower than SRAM
  - Delay for L2 cache consists of logic/wire/array delay; array access is small component
- eDRAM creates new, favorable design points with respect to SRAM cache
  - Constant capacity (MB) at smaller area – potentially faster due to shorter wire delay
  - Constant size (area) – performance improvement from larger (4X) capacity

IBM Research

## Integration - Supercomputing example

	No.	Date	Tera flops	Features
NEC Earth Simulator	1	Nov/2002	36	Specialized vector machine, leading-edge technology
IBM BlueGene/L Beta	1	Nov/2004	71	Massively parallel, based on <b>SoC</b> technology from embedded systems ( <b>PPC 440</b> )
SGI Columbia	2	Nov/2004	52	512-nodes clusters ( <b>Itanium2</b> )
IBM MareNostrum	4	Nov/2004	21	Blade-server, based on desktop processors ( <b>PPC 970</b> )
IBM BlueGene/L eServer	1	Nov/2005	281	131072 BlueGene processors
IBM BlueGene/W eServer	2	Nov/2005	91	40960 BlueGene processors
IBM ASC Purple eServer	3	Nov/2005	63	10240 POWER5 processors

Leadership achieved through "non-traditional" means

9 J. Moreno / May 2006 © 2006 IBM Corporation

IBM Research

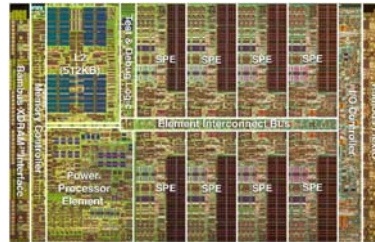
## BlueGene – an example of chip/system level integration

Type of integration	Coupling to CPU	Features
Multi-core	Decoupled	Dual-core on a single chip, each core operating independently, potentially on different functionality (compute, communicate). Identical cores
Dedicated units	Loosely-coupled	Specialized units to accelerate inter-processor communication
Instruction set extensions	Tightly-coupled	Two-wide SIMD double-precision floating point instructions

10 J. Moreno / May 2006 © 2006 IBM Corporation

## Cell – another example

- Multiprocessor on a chip
  - 241M transistors, 235mm<sup>2</sup>
  - 200 GFlops (SP) @3.2GHz
  - 200 GB/s bus (internal) @ 3.2GHz
- One general-purpose core
  - 64-bit Power Architecture™ with VMX (SIMD engine)
  - General-purpose processor
- Eight specialized cores
  - 128-bit processors, SIMD
  - Higher computing density and better efficiency
  - Data-level parallelism, instruction-level parallelism, task level parallelism
  - DMA controller



## Key transitions in systems

- 1980s : Bipolar technology
  - Multi-chip modules, frequency improvements
- 1990 – 2004: CMOS technology
  - Frequency and density
  - Single-chip processor (Power) and multi-processor chips (Power4)
- 2005 and beyond: CMOS Systems-on-a-chip (SoC)
  - Integration and density
  - Multicore processor chip with additional functionality, developed using ASIC methodology: BlueGene processor

## Technology implications

- Power consumption and power density became a disruptive factor
- Technology progress will continue, but not through scaling as we know it
- Integration becomes a source of differentiation
- New technologies offer opportunity to leverage integration and enable innovation through alternative dimensions

Opportunities -or demand- for alternative forms of improving performance

## Integration and modularity

- Increased reliance on modular components
  - Configurability emerging as possible replacement for traditional design
  - Systems built using –potentially suboptimal- components (modules)
    - Remember MSI/LSI-days ...!!!
- Software enablement becoming ever more important in exploitation of modularity and configurability (compilers, operating systems, middleware, etc.)
  - Need to hide the hardware complexity

## Chip-level integration opportunities

- Adding functionality to the processor core
  - New instructions and functional units, used selectively
- Multiple cores integrated at the chip level
  - Same instruction-set architecture but potentially different microarchitectures: asymmetric multiprocessor
  - Different instruction-set architectures (for different domains or different components within the same domain): heterogeneous multiprocessor
- Enhanced memory hierarchy: caches, eDRAM, enhanced functionality, etc.
- Accelerators/off-load engines : fixed-function, programmable, etc.
- Chip interconnect fabric
  
- Multiple hardware mechanisms for enhancing performance and efficiency throughout the entire chip

## Application-optimized systems

- Application optimized systems getting widespread adoption
  - Alternative for improved performance given technology restrictions
  - Large performance gains (10X or more) enabling new classes of applications
- Modularity at all levels is key element
  - Chip-level: multiple cores, multithreading, specialized cores, accelerators, interconnects
  - Subsystem-level: memory, I/O, specialized blades
  - System level: blades, drawers, racks
  - Business level: business processes
  
- New and evolving workloads suitable for application-optimized systems

## Software challenges in multicore systems

- Many independent threads
  - Number of threads in the chip vs single-thread performance
- Many MPI threads (as in BlueGene)
  - Number of threads vs efficiency of messages
- Moderately-threaded applications
  - Number of hardware threads vs number of software threads
- Single-threaded application
  - Use of several hardware threads on behalf of single task vs forced migration into threaded version (re-program)
- Transition to multiple cores should be as transparent as possible
  - Reduce impact to users/developers
  - Inertia prevents adoption of new architecture features unless very large gains perceived
- Performance sensitive applications will likely become (or have already become) candidates for threaded implementations
  - Others need quasi-automatic exploitation of multiple cores, otherwise will remain single-threaded
- Dynamic execution environments
  - Can enable transparent exploitation of multiple threads
  - Might not achieve best performance but provides path to ease migration into multi-core chips

## Hiding complexity: “Single source” compiler for Cell

- The user prepares an application as a collection of one or more source files containing user directives targeting the Cell architecture
- Guided by these directives, the compiler
  - takes care of partitioning the code between PPE and SPES
  - spreads the parallel code across all SPES available
  - takes care of data transfers.
    - identifies accesses in SPE functions that refer to data in system memory locations
    - uses software cache or static buffers to manage transfer of this data to/from SPE local stores
  - takes care of code size
    - partitions code into manageable sizes and effects the transfer in and out of local store
- Eventually, directives replaced by compiler algorithms

[www.research.ibm.com/cellcompiler/compiler.htm](http://www.research.ibm.com/cellcompiler/compiler.htm)

## Development and analysis tools

- Monitoring and analysis tools for detecting thread exploitation opportunities
- Thread-generation tools
  - Static, post-link, dynamic
- Compiler optimization technology
- Performance evaluation and tradeoff analysis tools
- Runtime support
  
- Need to improve facilities over state of the art for single thread applications

## Concluding remarks

- Chip-level integration is a new frontier for microprocessor design
  - “Chip-level architecture” as opposed to “core architecture”
  - Parallelism and function tuning as alternatives to scaling
- Challenges in the evolution of architected functionality
  - WHAT – driven by emerging areas of computing
  - HOW - Conceive and evaluate alternative forms of integrating new functionality, the evolution of such functionality, and the associated software enablement needed for widespread adoption in general-purpose platforms (servers, desktop, etc.)
    - Features, programmability, resource management, etc.
    - Implementation alternatives
    - Programming models, software development tools, etc.....

## Concluding remarks

- Core-level architectures were developed about 30 years ago, when technology and applications were drastically different to the current ones.
- Combination of chip-level architecture and enablement environment (ecosystem) for multicore architectures can hide the complexity of heterogeneous multi-core chips
- SoCs already have integrated features
  - However, embedded systems are mostly application-specific
- Lessons learned in SoCs can lead to generalizations that makes them applicable to general-purpose server/desktop systems
- Hide complexity: make a multi-core chip appear as if programming a single processor

