technology
from seed

Foreseeing the Role of Reconfiguration in Multi-core Architectures

Leonel Sousa
with
Pedro Trancoso, Frederico Pratas and Panayiotis Petrides

Technical University of Lisbon/INESC-ID, Lisbon, Portugal

16th December, 2009

Motivation

- Shift in processor architecture towards multiple cores
  - to avoid the power and complexity walls
    - #Cores have been increasing for different types of devices
- Increasing number of cores leads to three major challenges:
  - core configuration and their memory hierarchies
  - management of such complex hardware
  - programmability for such systems

Motivation

- Back to the Amdhal law, efficiency with multi-cores depends on the fraction of the application which exibhits parallelism
- Which type of parallelism?
  - ILP, TLP, DLP, or all of them?
- Should this parallelism be identified and exploited statically or dynamically?
  - At compiler, OS, or hardware level?
- What is the role of the architecture/hardware?

Motivation

- Contemporaneous devices tackle some of these challenges by:
  - Having different architectural paradigms (e.g., GPP are homogeneous while the IBM Cell is heterogeneous)
  - Supporting different memory organizations (e.g., GPP support shared memory while IBM Cell does not).
  - more general purpose or more specialized (GPU)
  - Different hardware granularity and amount of parallelization (GPP vs GPU)

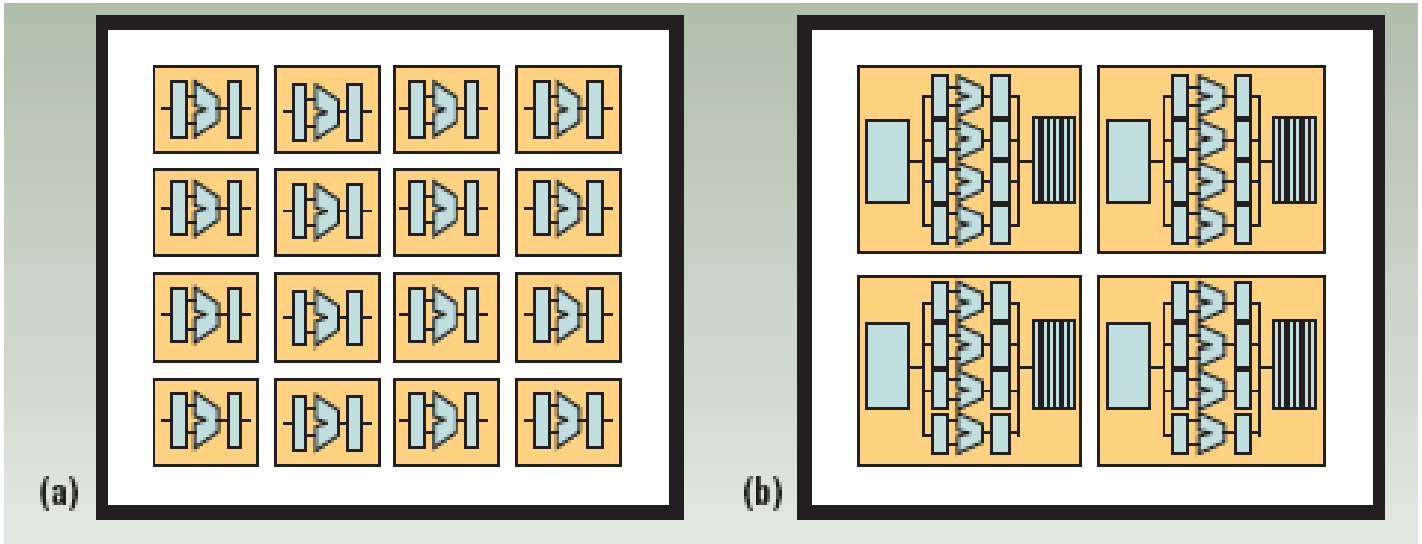
Motivation

- However future processors will probably have to offer cores which combine several of these specifications
  - which could even change dynamically at run-time.
- One solution is for future Multi-core architectures to be intrinsically heterogeneous and configurable
  - to achieve a better match between applications and the hardware
  - to dynamically adapt the hardware to the application needs
- In the last years research has been done to introduce heterogeneity and configurability in multi-cores
  - but is has not yet reached the mass market

Outline

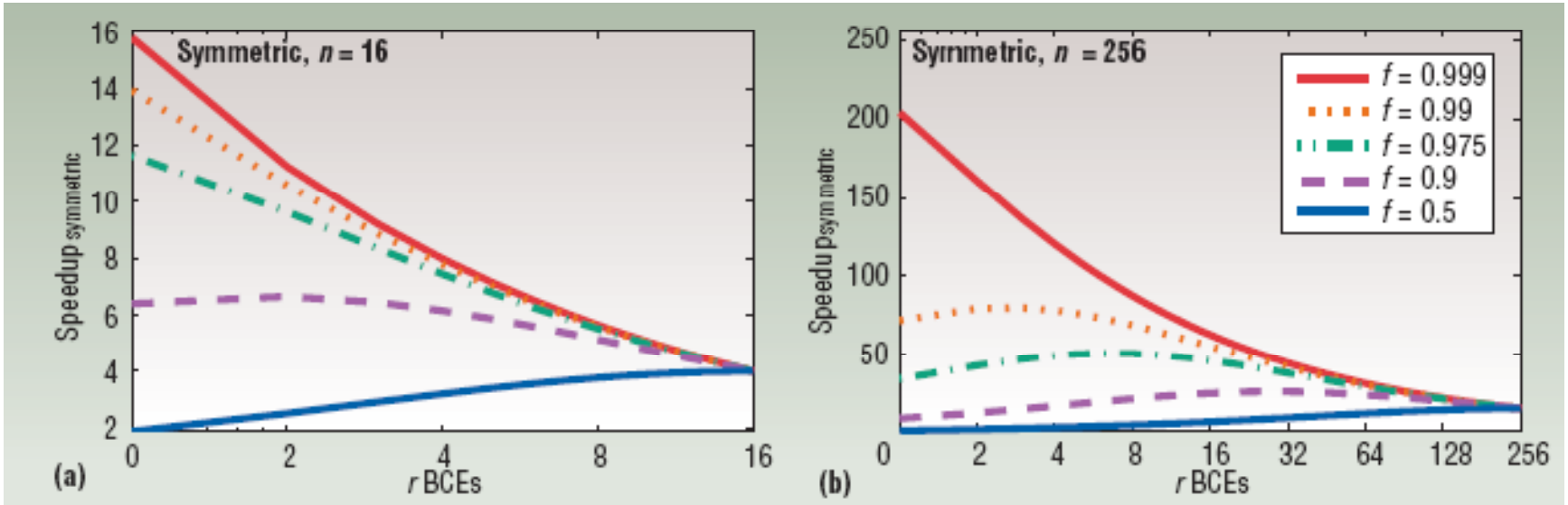
- Current multi-core architectures with different characteristics
- Extension of Amdahl's law for multi-cores
- Research on Reconfigurable Architectures
- Future of multi-core architectures: heterogeneous and reconfigurable?

Multi-cores: extension of Amdahl's law



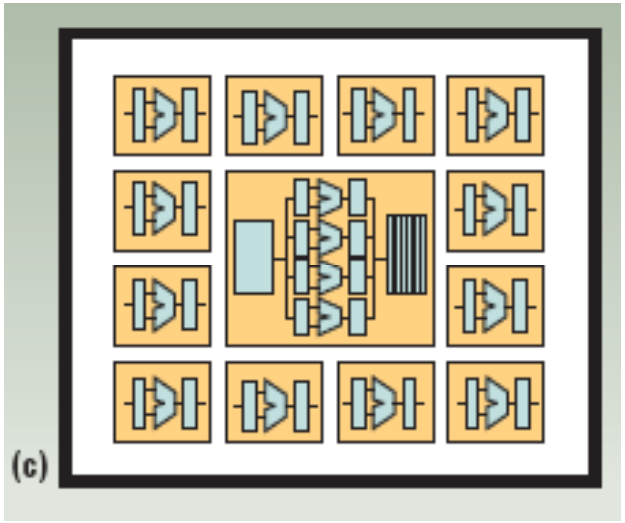
$$\text{Speedup}_{\text{symmetric}}(f, n, r) = \frac{1}{\frac{1-f}{perf(r)} + \frac{f \cdot r}{perf(r) \cdot n}}$$

Multi-cores: extension of Amdahl's law



- finding parallelism is critical (f near 1)
- Using more BCEs per core, r > 1, can be optimal, even when performance grows by only SQRT(r)
  - *for n = 256 and f = 0.975, the* maximum speedup occurs using 7.1 BCEs per core

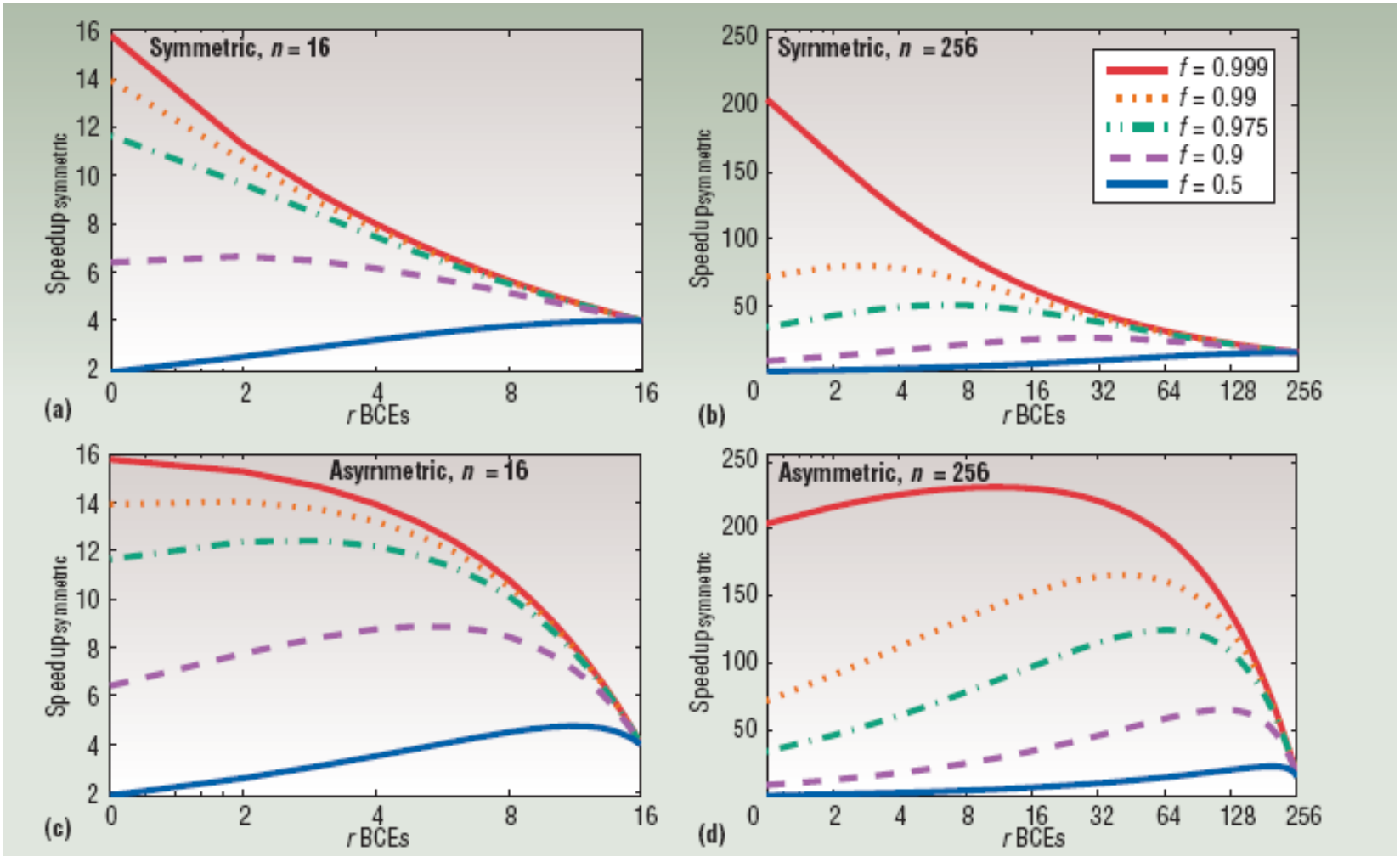
Multi-cores: extension of Amdahl's law



$$\text{Speedup}_{\text{asymmetric}}(f, n, r) = \frac{1}{\frac{1-f}{perf(r)} + \frac{f}{perf(r)+n-r}}$$

- Asymmetric multicore offer potential speedups much greater than symmetric
  - *f = 0.975 and n = 256, the best asymmetric speedup is 125.0, whereas the best symmetric speedup is 51.2*
- Denser multi-core increase the speedup
  - going asymmetric increases performance

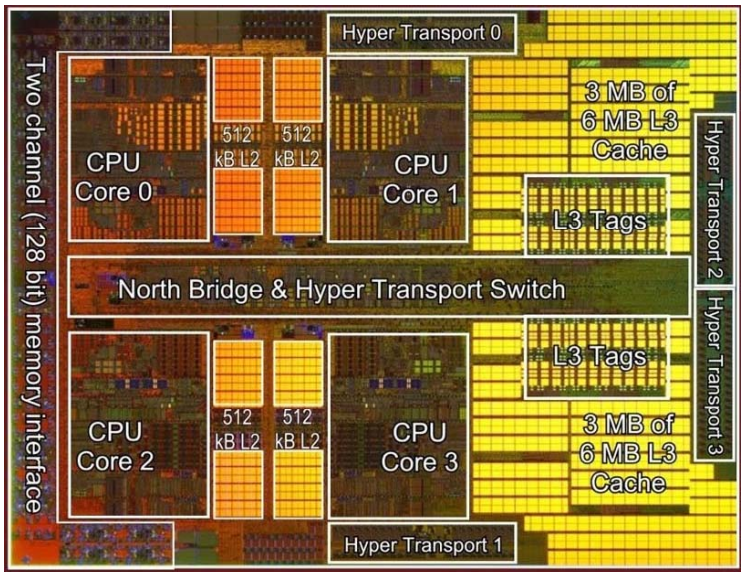
Multi-cores: extension of Amdahl's law



General-Purpose Homogeneous Multi-Cores: AMD Quad Core Shangai

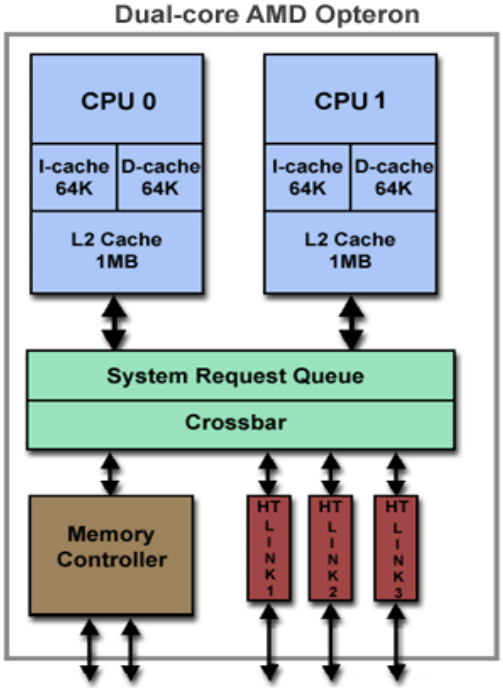
- Small number of **identical** cores
- Complex cores able to **efficiently** exploit Instruction Level Parallelism (ILP)
- Multi-level **hardware managed cache** memory
- Typically hardware supports memory **coherency**
- **Shared cache** → Efficient data transfer and synchronization

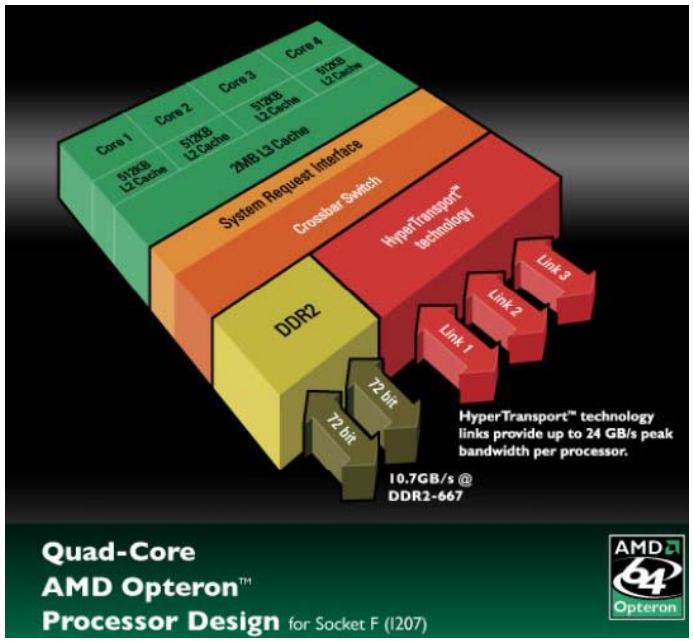
AMD Quad Core Shanghai



General-Purpose Homogeneous Multi-Cores
Opteron

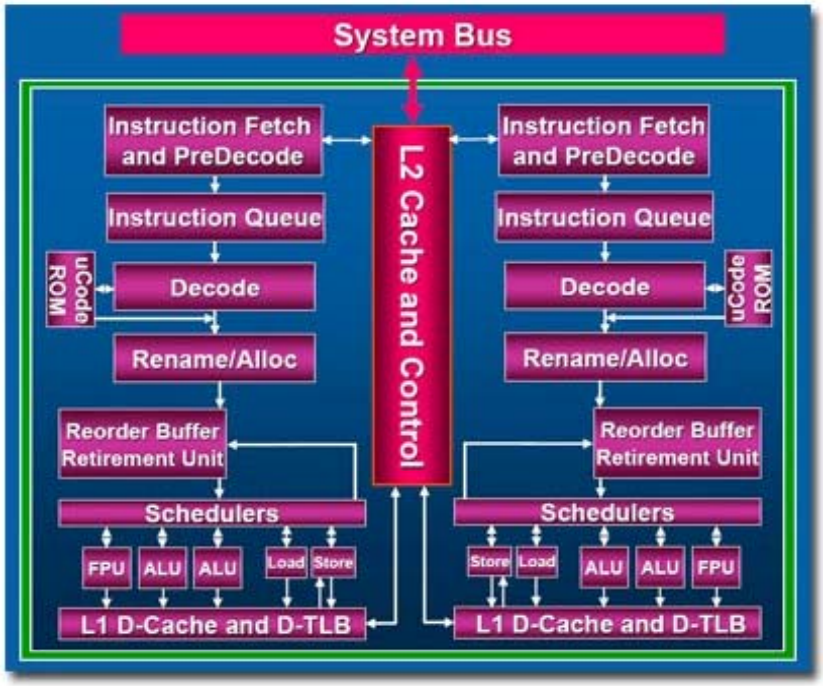
- **Dual-core and Quad-core AMD Opteron**
  - Each processor's core execute up to **3 instructions per cycle**
  - **Individual L2 caches**
  - Data used by both cores is shared in the Main memory (NUMA)
  - A 128-bit SSE instructions can be executed per cycle





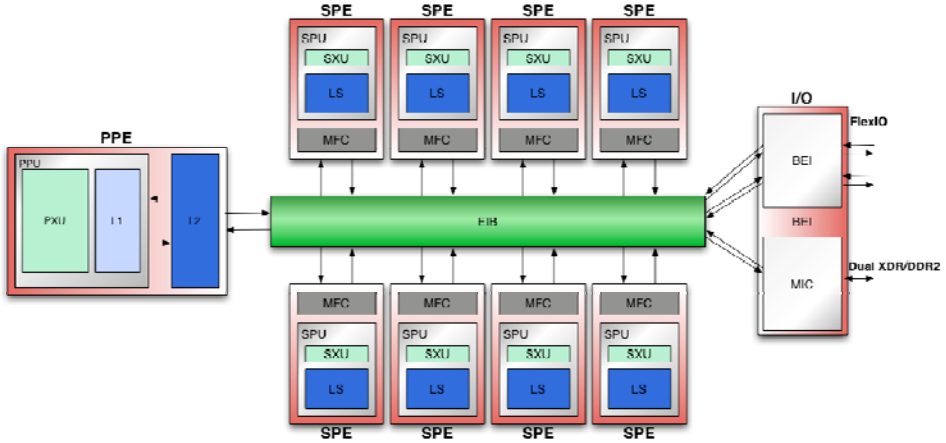
General-Purpose Homogeneous Multi-Cores: Core 2 Duo

- **Intel Core 2 Duo** run at lower frequency than Pentium 4
  - Each processor's core execute up to **4 instructions per cycle** and 24 GFLOPS@3GHz
  - **Shared L2 cache**: same copy of data used by both cores and more heavily loaded core can use bigger portion of L2 cache
  - A 128-bit SSE instructions can be executed per cycle



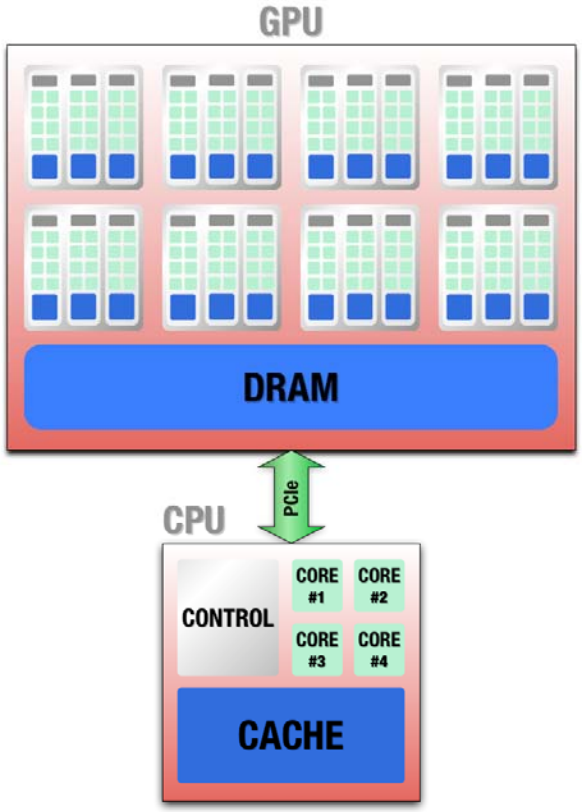
Heterogeneous Multi-core: IBM Cell/BE

- Heterogeneous Multi-core architecture (9 cores):
  - **1x general-purpose** PowerPC Processor Element (PPE)
  - **8x special-purpose** Synergistic Processing Elements (SPEs)
- SPEs include a private unified memory with 256KB.
- PPE and SPEs **communicate** through the Element Interconnect Bus (EIB) *via* **DMA** data transfers.
- Software managed cache → **user** is **responsible** for efficiently manage the **memory** space.
- Programmed using the Cell/SDK (thread-based) library extensions



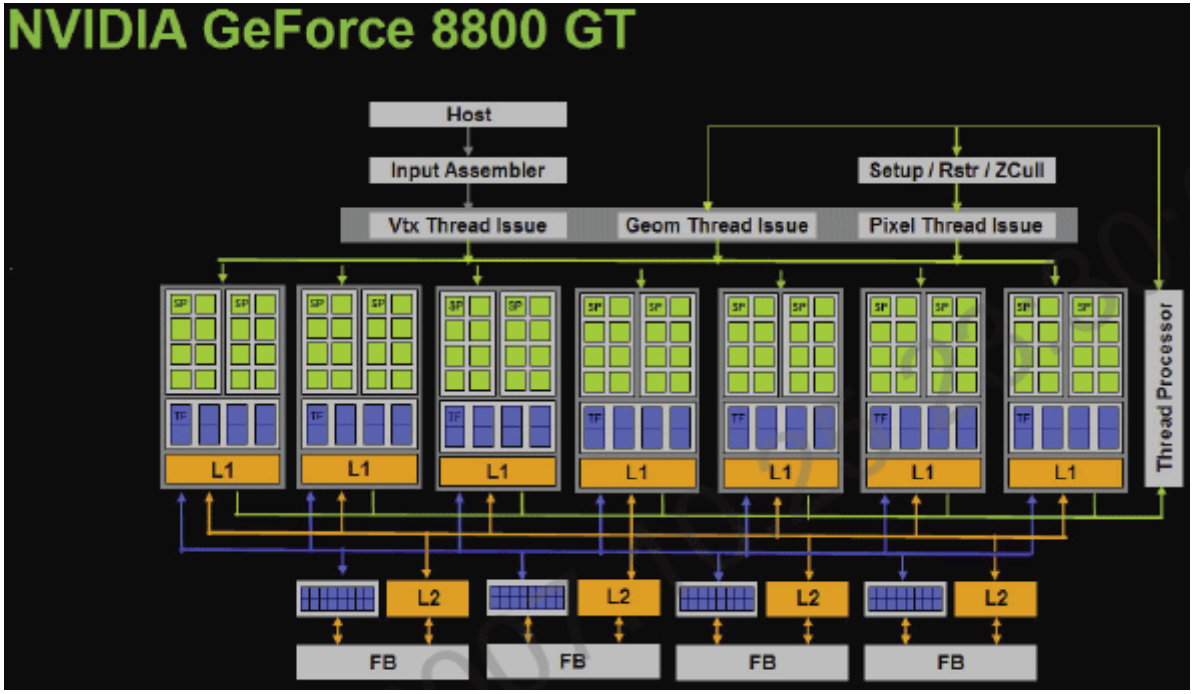
Graphics Processing Unit (GPU)

- Acceleration unit → heterogeneous system
- Can be used for general-purpose processing (GPGPU)
- Host and GPU are usually connected through a system bus (e.g. PCIe)
- Includes a large number of very **basic** processing **cores** (1000s) → highly **multi-threaded**
- Data is organized **hierarchically** in memory and **managed** by the **user**



Graphics Processing Units (GPU)

- TESLA architecture: 128 stream processors clocked at 1.35 GHz
  - 576 Gflops and 86.4 GB/sec of memory BW
- Data parallelism -> the same program runs on all processors, otherwise the performance becomes quite low

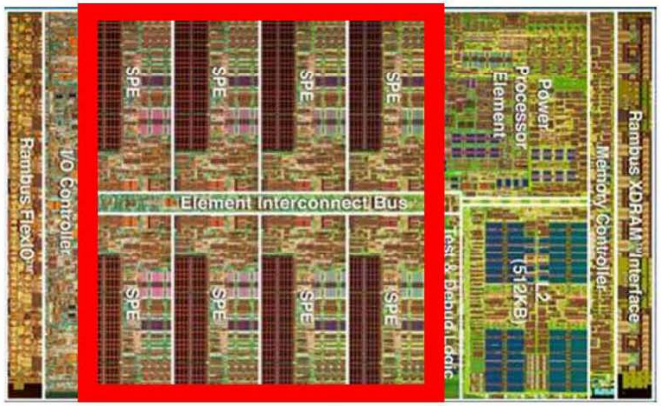


Evaluation: Layout

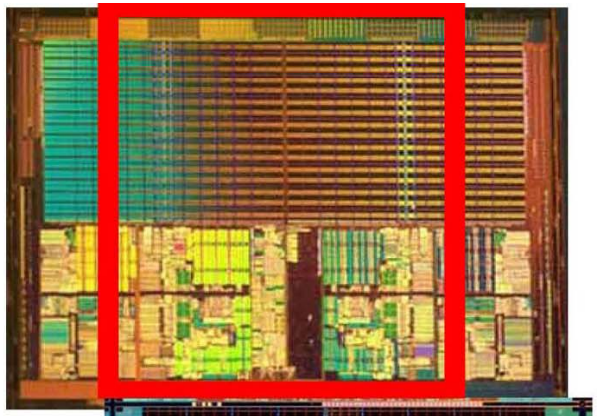
technology
from seed

inesc id lisboa

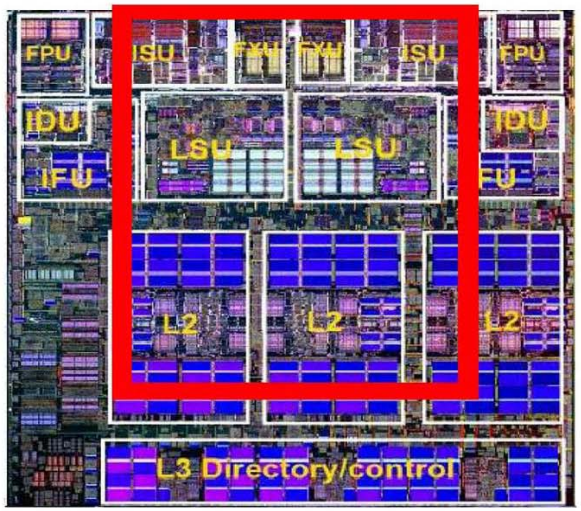
Cell BE



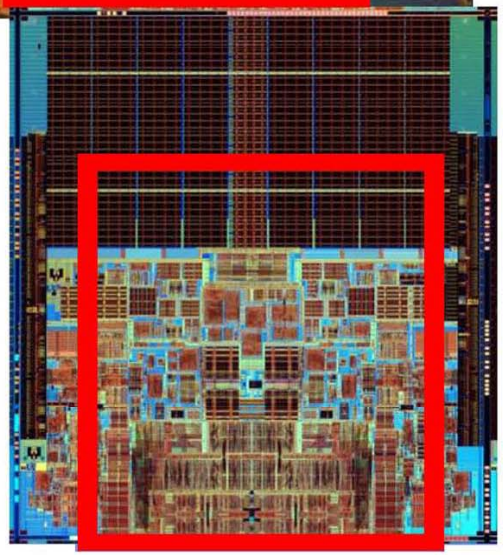
AMD



IBM

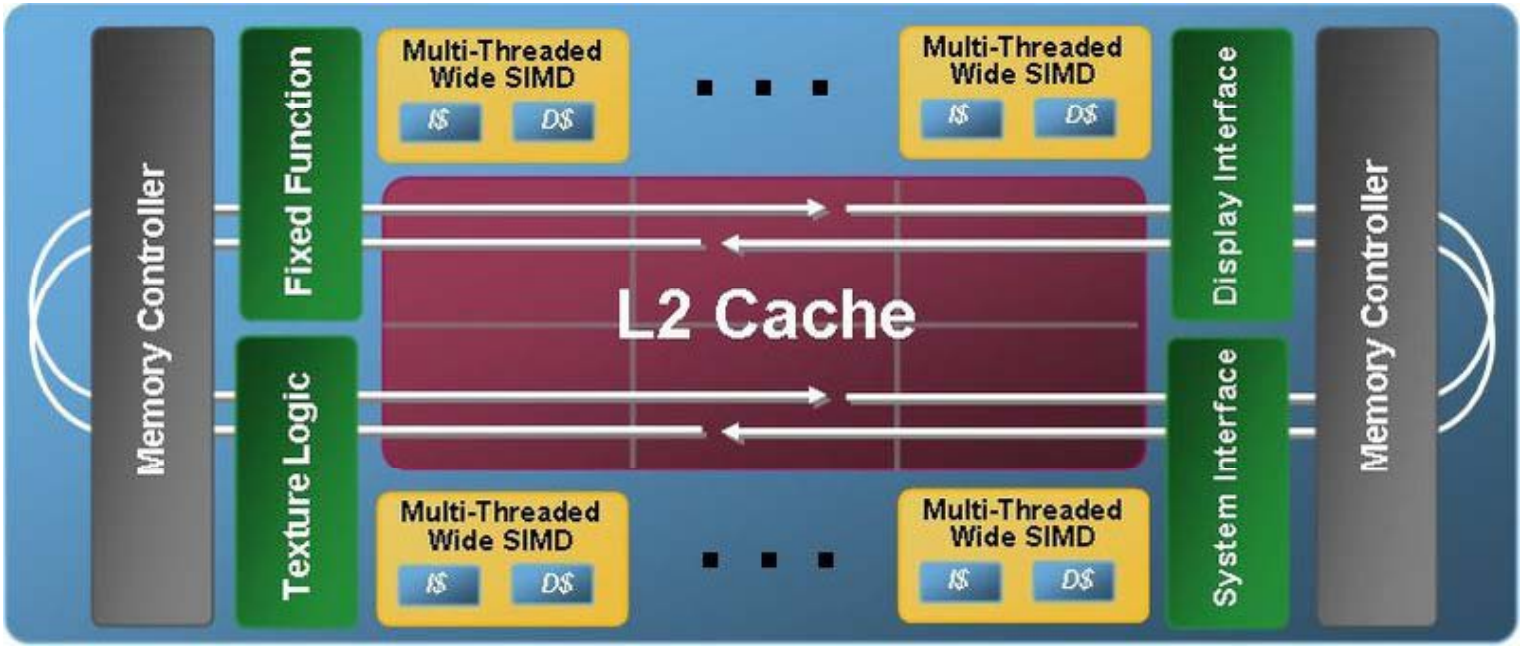


Intel



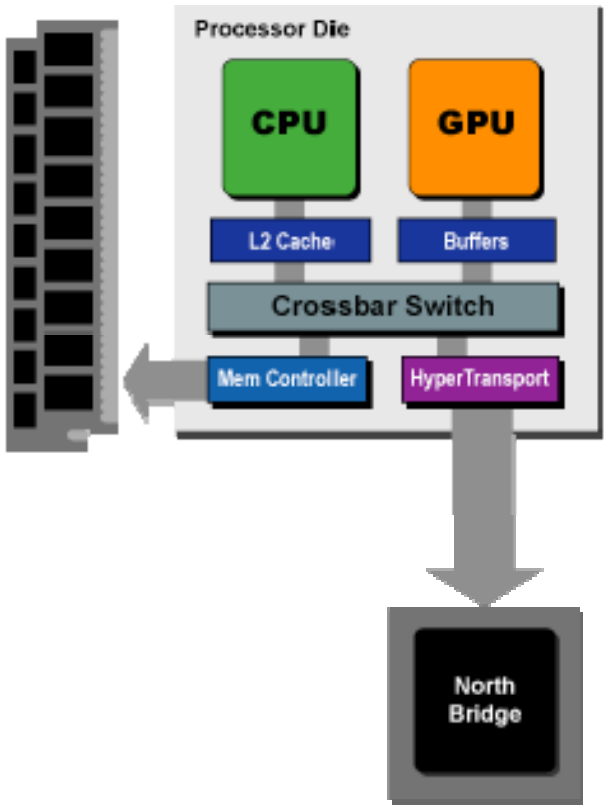
INTEL Larrabee GPU/CPU

- Shared memory architecture through L2 cache
- x86 ISA with wide SIMD vector instructions – 512 bits



AMD Fusion

- Moving the graphics processor from the northbridge to the CPU
  - graphics processor will be able to access the backside pool of main memory directly
    - lower latency for GPU main memory accesses
- In Intel chipset IG, memory controller and GPU are already on the same die
  - a more integrated part would have multiple SIMD stream processors (i.e. unified vertex/pixel pipelines) that share an on-die bus with a GPP, much like Cell's architecture



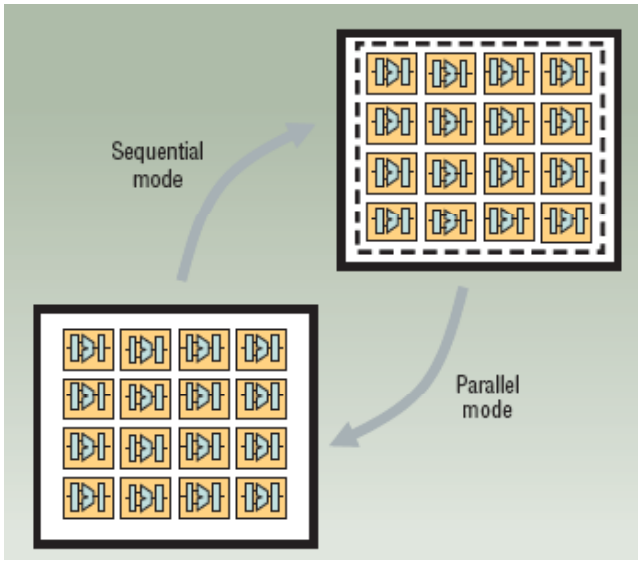
Brief Comparison

technology
from seed

inesc id lisboa

	Homogeneous	Heterogeneous	Accelerators
Main Characteristics	small #cores HW managed memory	small #cores limited SW managed memory	large #cores SW/HW managed memory
Disadvantages	Hidden overheads	Execution in smaller steps Synchronization Programmability	Global data transfers Programmability
Advantages	Fast communication via shared cache	Concurrent communication/ computation	High throughput

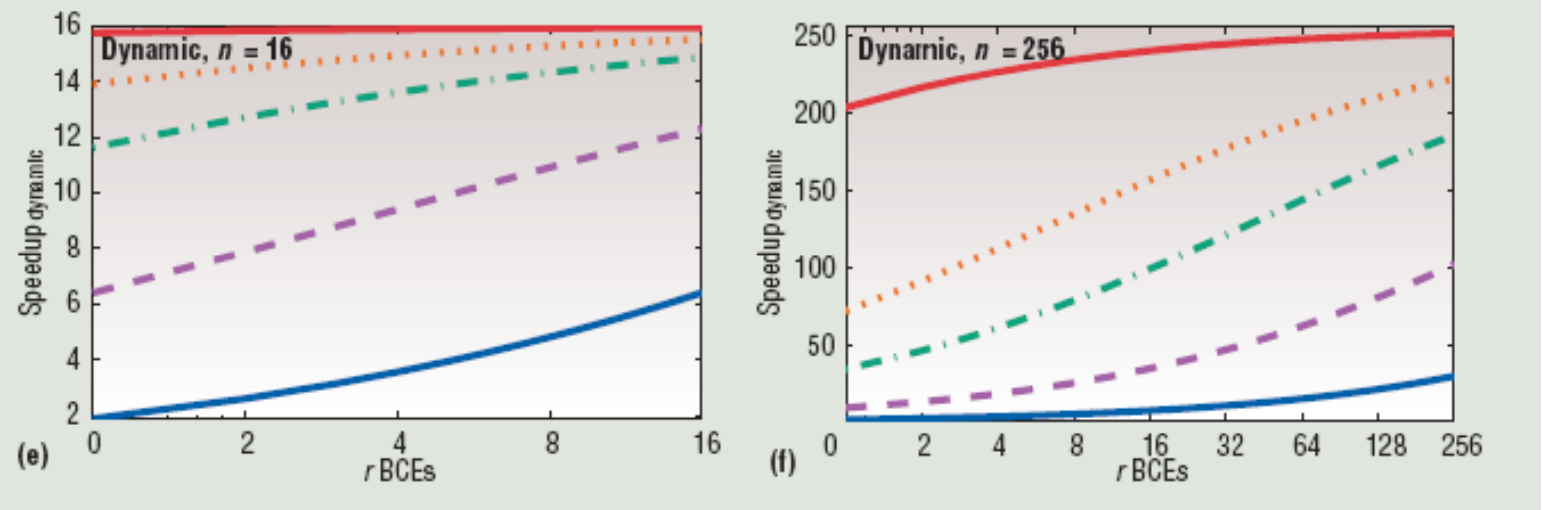
Extension of Amdahl's law to Reconfigurable Multi-cores



- Dynamic multi-cores can offer speedups that can be greater (and are never worse) than asymmetric chips with identical *perf(r) functions.*

$$\text{Speedup}_{\text{dynamic}}(f,n,r) = \frac{1}{\frac{1-f}{perf(r)} + \frac{f}{n}}$$

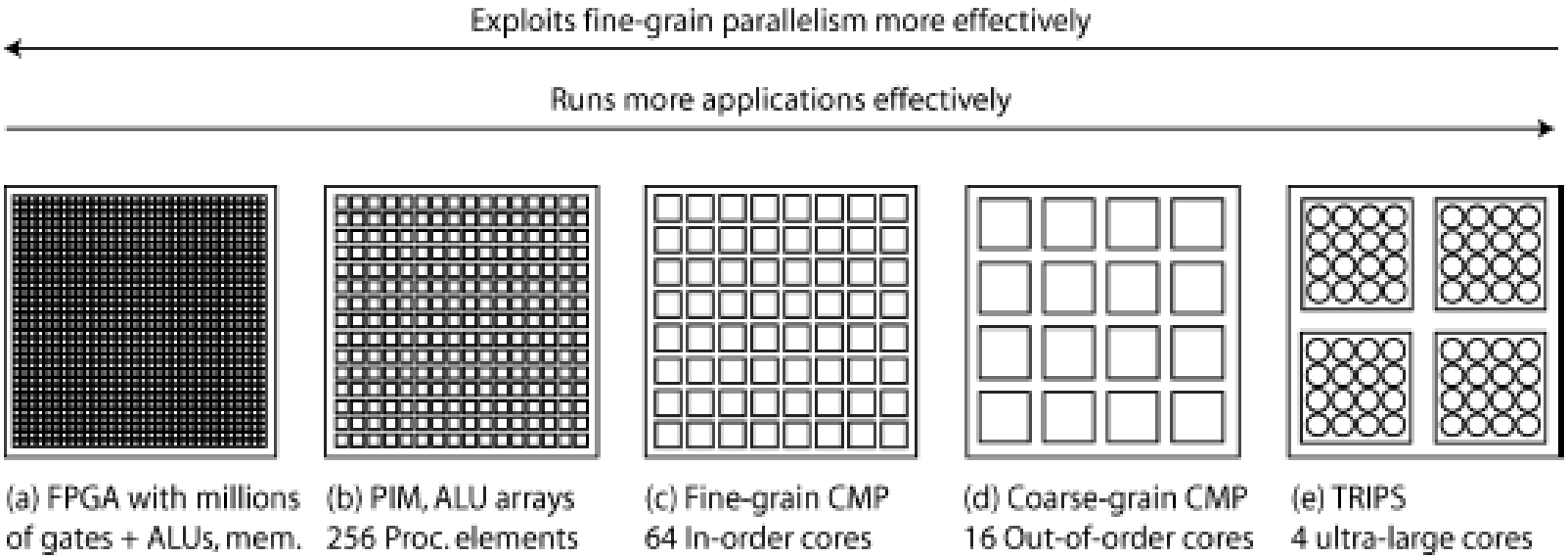
Multi-cores: extension of Amdahl's law to reconfigurable



- Dynamic multi-cores can offer speedups that can be greater (and are never worse) than asymmetric chips with identical *perf(r) functions.*
- Amdahl's sequential-parallel assumption achieve greater speedup than asymmetric chips but requires reconfiguration
  - More cores for sequential mode than is possible today.

Reconfigurable Multi-cores

- Architectural polymorphism
  - capability to configure hardware for efficient execution across broad classes of applications
- Granularity?? For a given granularity which architecture??



Reconfigurable Multi-cores

- Finer-grained architectures offer high performance on applications with fine-grained (data) parallelism
  - difficulty achieving good performance on general-purpose and serial applications
  - e.g, the performance of PIM topology on control-bound codes with irregular memory accesses, such as sparse matrix scientific codes or program compilation, would be bad

- coarser-grained architectures have not had the capability to use their internal computational resources to show high performance on fine-grained, highly parallel applications

- Polymorphism bridge this dichotomy with either of two competing approaches
  - synthesizing approach uses fine-grained CMP to exploit applications with fine-grained, regular parallelism, and tackles irregular, coarser-grain parallelism by synthesizing multiple processing elements into larger “logical”

    CORE FUSION

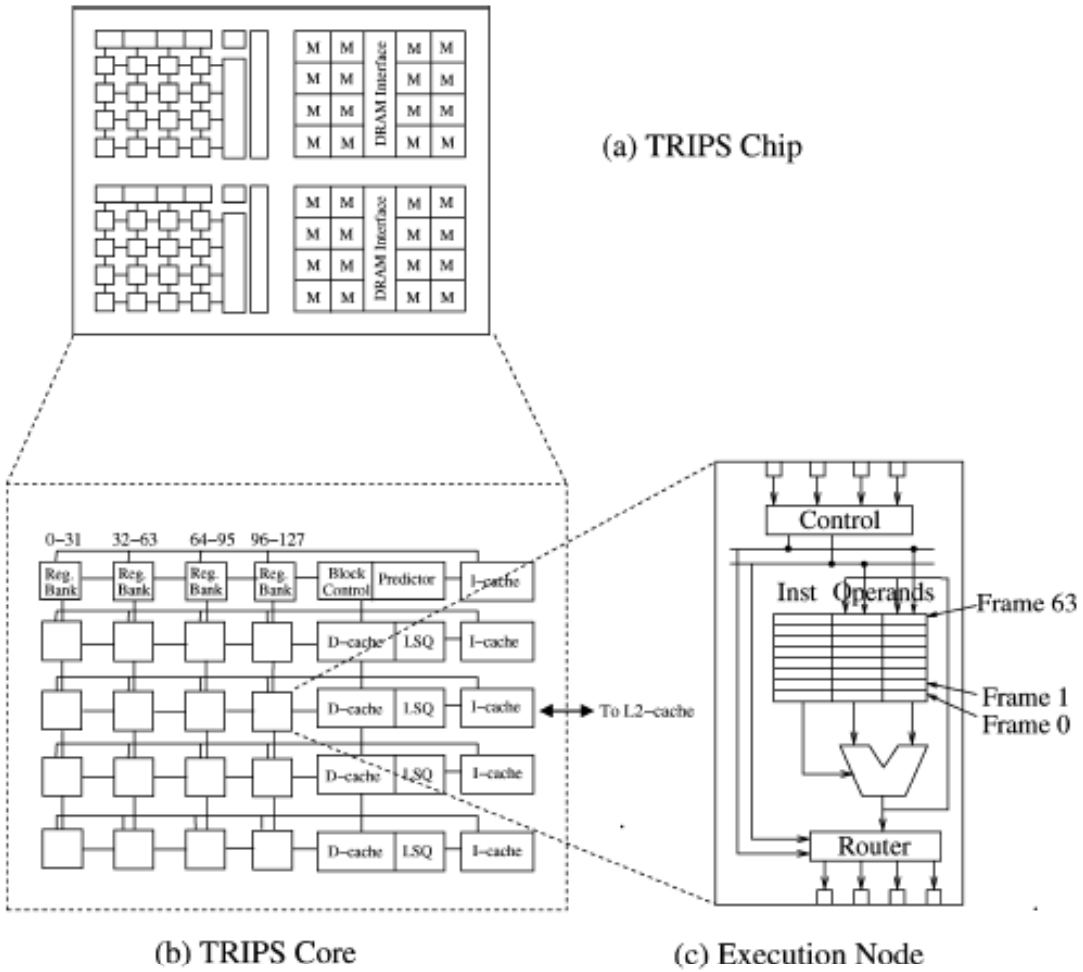
  - partitioning approach implements coarse-grained CMP, and provides configuration and ISA support to partition the large processors logically, exploiting finer-grain parallelism when present

    TRIPS

Reconfigurable Multi-cores: TRIPS

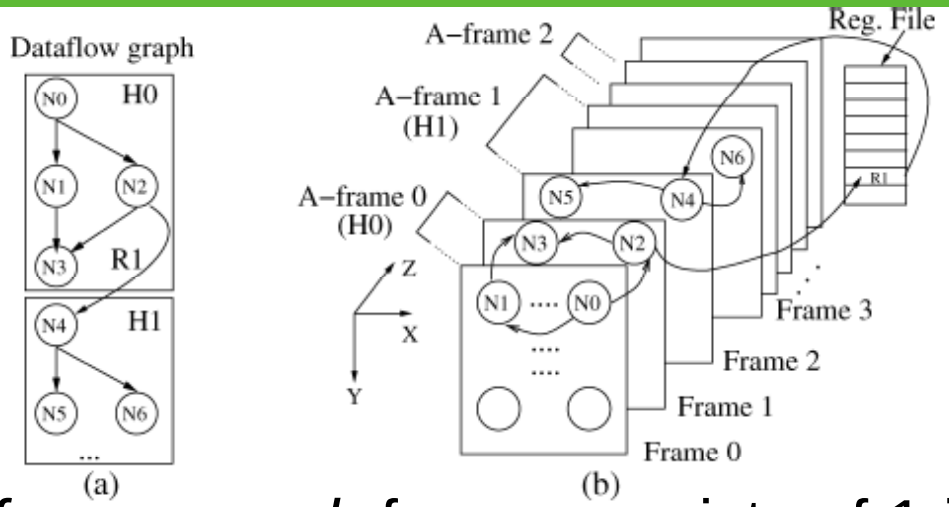
- TRIPS architecture employs large, coarse-grained grid processors (GPA) to single-threaded applications (ILP)
- Explicit data-graph execution (EDGE) instruction set, augmented with polymorphous features to subdivide core
- Partitioned computation and memory elements are connected by point-to-point communication channels that are exposed to software schedulers
  - TRIPS processor cores and memory system are essentially pools of distributed ALUs and memory banks

Reconfigurable Multi-cores: TRIPS



Prototype:
4 × 4 array of ALUs with 64 reservation stations per ALU, thus has 64 frames of 16 instructions each

Reconfigurable Multi-cores: TRIPS



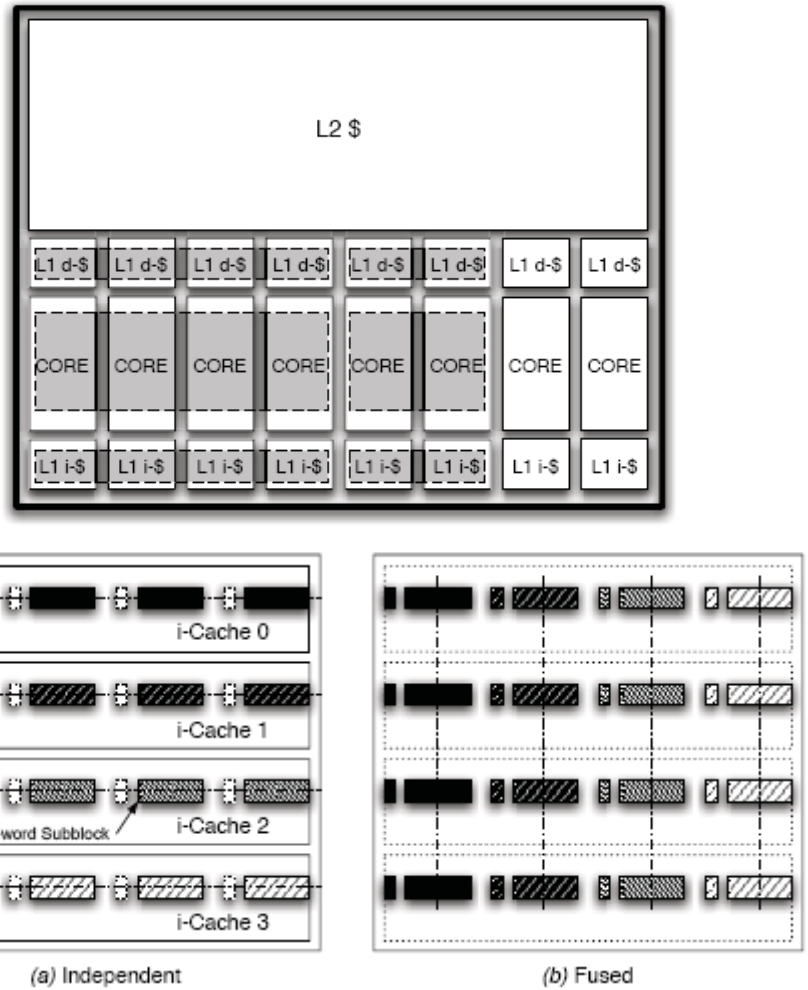
- Series of *frames, each* frame consists of 1 instruction buffer entry per ALU node: 2D slice of the 3D scheduling region
- compiler schedules hyperblocks 3D region
  - assigning each instruction to one node in the 3D space
- H0 mapped into A-frame 0
  - instructions N0 and N2 are mapped to different buffer slots (frames) on the same physical ALU node.

Reconfigurable Multi-cores: Core Fusion

- Empowers groups of relatively simple and fundamentally independent CMP cores with the ability to “fuse” into one large CPU on demand

- CMP as a homogeneous substrate with conventional memory coherence/consistency support, optimized for parallel execution, but where groups of up to four adjacent cores and their i- and d-caches can be fused at run-time:
  - fetch, issue, and commit width, and up to four times the i-cache, d-cache, branch predictor, and BTB size.

Reconfigurable Multi-cores: Core Fusion



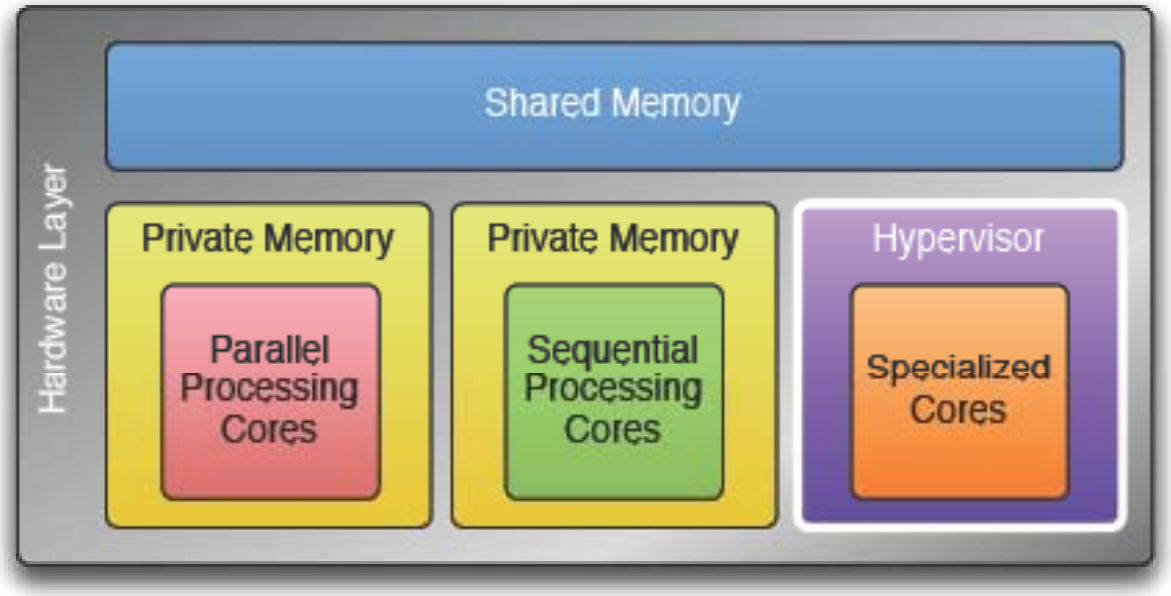
- As different applications, and even different phases of the same application, have different demands, a processor with a diversity of cores would be able to achieve a better application-to-hardware match, resulting also in better power-performance efficiency

- **Team of cores**:
  - core elements with complementary skills which generate synergy through a coordinated effort allowing each element to maximize its strengths and minimize its weaknesses

Future of multi-core architectures: heterogeneous and reconfigurable?

- Manufacturers in the future provide not only hardware platform but also a virtualization layer for that platform that hides the complexity and diversity of the hardware.
  - operates as autonomic manager of the underlying Team-of-Morphable cores, releasing programmer from this demanding task.
- Offering this virtualization layer along with the hardware it is possible to offer a standard set of core services to the upper layers
  - run the same application on different hardware multi-core platforms
- The virtualization layer may provide a number of virtual cores for the execution.
  - A regular OS may do task scheduling on the virtual cores.

Future of multi-core architectures: heterogeneous and reconfigurable?

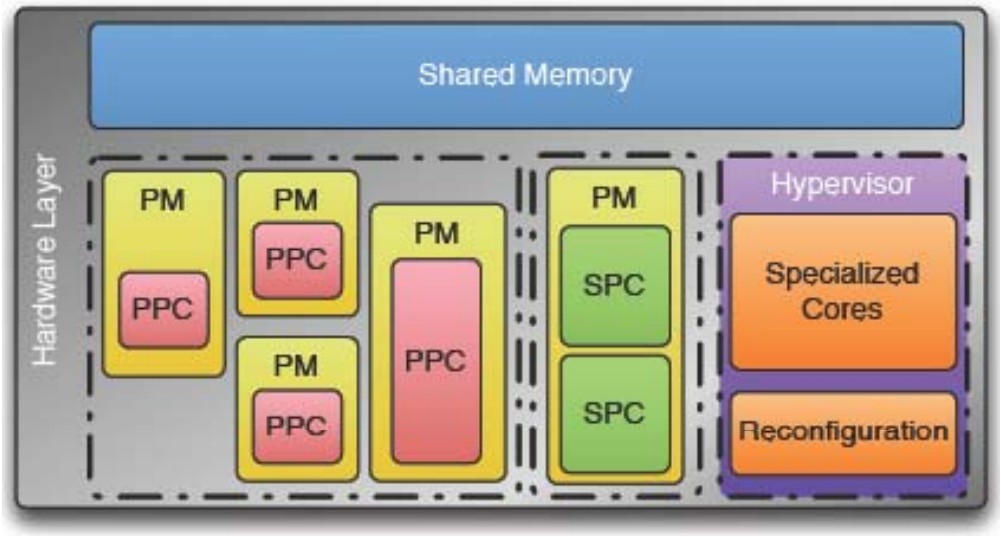


- the cores implemented share the same baseline ISA, which can be any canonical ISA such as x86
  - in addition, some cores may also include ISA extensions, e.g., for multimedia (such as MMX/SSE, AltiVec and cryptography)

Future of multi-core architectures: heterogeneous and reconfigurable?

- The architecture consists of several cores with different functionalities which combined in a team are able to improve the overall system performance
- Since Hypervisor is general to manage architectures with different characteristics, there is not a strict definition of how the processor cores should be configured
  - Considering in a broad sense the requirements of actual applications, we can divided the cores into: general purpose cores for (a) parallel processing, and (b) sequential processing; the third group (c) specialized cores, is used to support the Mechanisms provided by the Hypervisor.

Future of multi-core architectures: heterogeneous and reconfigurable?



- Application behavior is unpredictable at hardware design time represents
  - not possible to define a priori the attributes of the several cores
- Reconguration is a way to surpass this limitation
  - naturally as a solution to dynamically reconstruct the architecture structure according to each application requirements

Future of multi-core architectures: heterogeneous and reconfigurable?

- It is possible to improve the overall performance and at the same time have a more efficient architecture
  - However, reconfiguration introduces a new level of complexity into the system. Once more, the Hypervisor can be used to encapsulate this additional complexity as a Service.
- Coarse-grain reconfigurable approach
  - allows to configure only the most relevant parts of the architecture; the architecture may include a discrete set of congurations in a limited reconfiguration space, thus reducing the design complexity and overheads
  - "morphable" designates those architectures which are not fully recongurable but are still able to adapt at run-time.

Future of multi-core architectures: heterogeneous and reconfigurable?

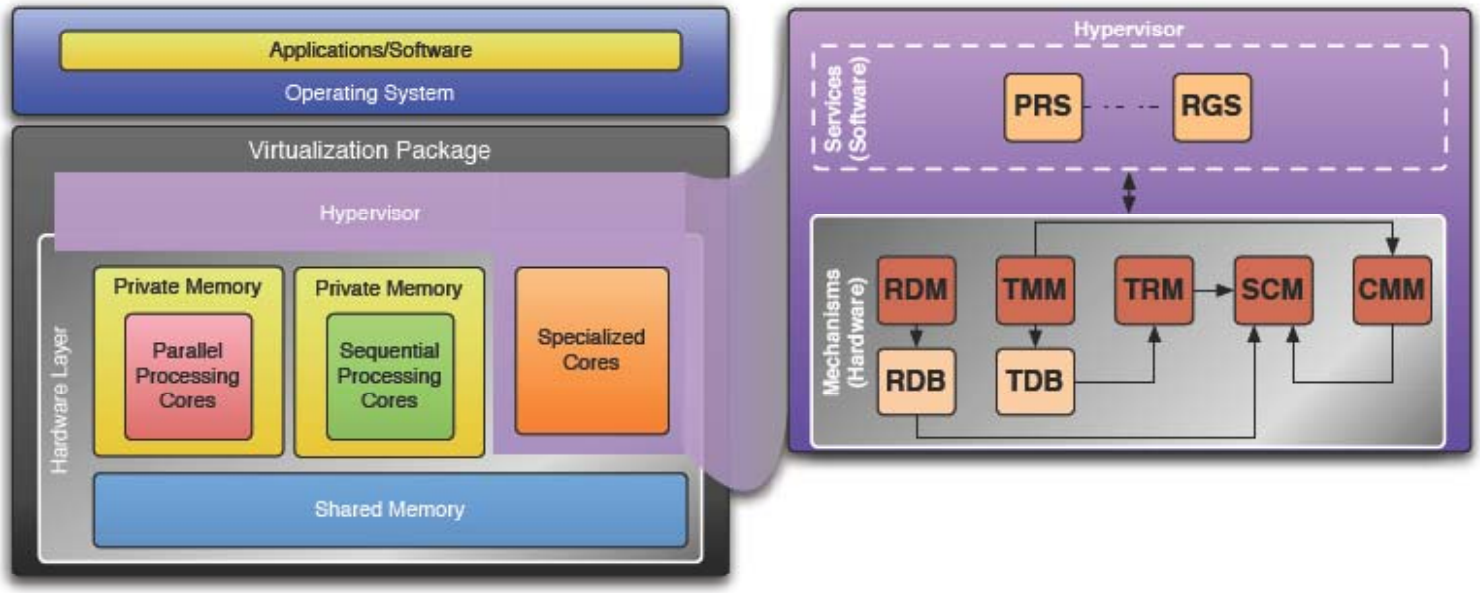


Figure 7. Overall Virtualization Package - Hypervisor Mechanisms and Services and underlying Hardware layer. Hypervisor Mechanisms can be ported on specialized cores whereas Services can be executed on Software supporting platform portability. System hardware layer might be composed of cores with different computation and memory characteristics.

Conclusions

- Future multicore processors will be composed of morphable cores with different computational and memory capabilities, which are able to change their configuration at run-time.
- Virtualization Package, i.e. a complete system able to wrap the complexity of the underlying hardware, through an Hypervisor module (hardware/software).
- The proposed team-of-morphable cores is composed of several asymmetric cores which have the capability of changing their configuration at different levels for both their logic and memory elements.

References

- Mark Hill and Michael Marty, ***Amdahl's Law in the Multicore Era***, *IEEE Computer*, July 2008, pp. 33-38

- Karthikeyan Sankaralingam, …, Doug Burger, *et al*, ***TRIPS: A Polymorphous Architecture for Exploiting ILP, TLP, and DLP***, *ACM Trans. On Architecture and Code Optimization*, vol. 1, nº 1, March 2004, pp.62-94

- Engin Ipek, *et al*, ***Core Fusion: Accomodating Software Diversity in Chip Multiprocessors***, International Symposium on Computer Architecture, California, June 2007, pp.186-197

- Panayiotis Petrides, Frederico Pratas, Pedro Trancoso, Leonel Sousa, ***Virtualization for Teams-of-Morphable Cores***, INESC-ID Internal Report, 2009, 22 pages

Questions

technology
from seed

inesc id lisboa

Questions?

Contact person: Leonel Sousa

las@inesc-id.pt