

Server Architectures

SMP: Symmetric Multi-Processors

January 2005

René J. Chevance

Foreword

- This presentation is an introduction to a set of presentations about server architectures. They are based on the following book:

Serveurs Architectures: Multiprocessors, Clusters, Parallel Systems, Web Servers, Storage Solutions
René J. Chevance
Digital Press December 2004 ISBN 1-55558-333-4
<http://books.elsevier.com/>

This book has been derived from the following one:

Serveurs multiprocesseurs, clusters et architectures parallèles
René J. Chevance
Eyrolles Avril 2000 ISBN 2-212-09114-1
<http://www.eyrolles.com/>

The English version integrates a lot of updates as well as a new chapter on Storage Solutions.

Contact: www.chevance.com

rjc@chevance.com

Organization of the Presentations

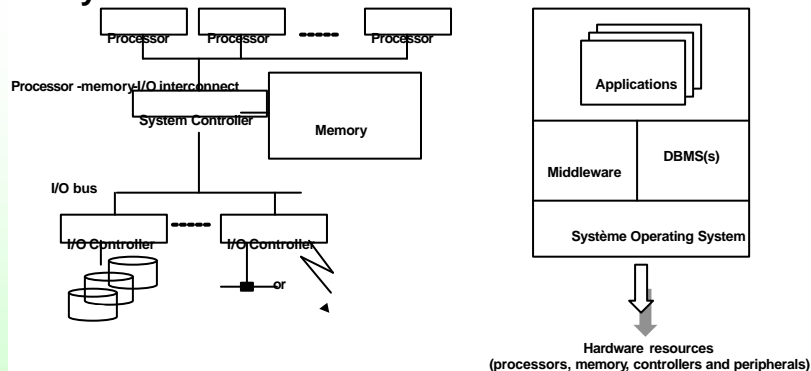
- Introduction
- Processors and Memories
- Input/Output
- Evolution of Software Technologies
- ➔ Symmetric Multi-Processors (this document)
 - SMP Architecture Overview
 - Microprocessors and SMP
 - Operating Systems and SMP
 - SMP with moderate number of processors (8 processors)
 - SMP with Many Processors (~16 processors): Nearly UMA Systems, CC-NUMA, COMA
 - Performance Improvement Provided by SMP
 - Advantages and disadvantages of SMP architecture
 - Partitioning: Virtual Machines, Hardware Partitioning, Logical Partitioning
- Cluster and Massively Parallel Machines
- Data Storage
- System Performance and Estimation Techniques
- DBMS and Server Architectures
- High Availability Systems
- Selection Criteria and Total Cost of Possession
- Conclusion and Prospects

Page 3

© R.J Cheavance

SMP Architecture Overview

■ Symmetric Multiprocessor Architecture



a) Hardware view of a Symmetrical Multiprocessor

b) Software View of a Symmetrical Multiprocessor

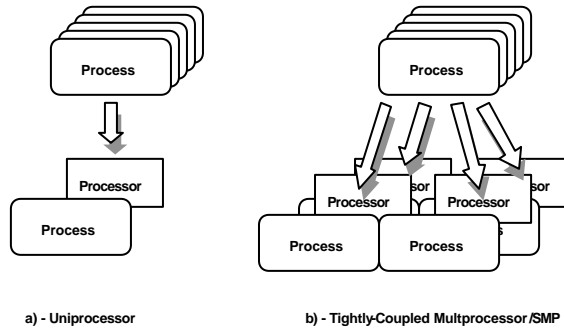
- With SMP, all processors may access to all system resources (memory, I/O). A SMP is operating under the control of one Operating System managing all system resources.
- The native programming model is the sharing of memory between processes.

Page 4

© R.J Cheavance

SMP Architecture Overview(2)

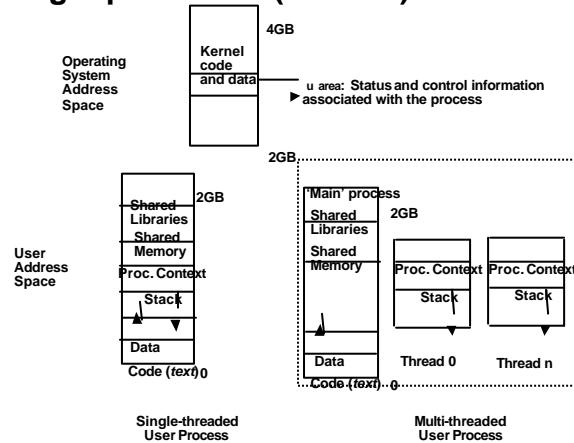
■ SMP Programming and Execution Model



Page 5
© R.J Chevrance

SMP Architecture Overview(3)

■ Lightweight processes (threads)

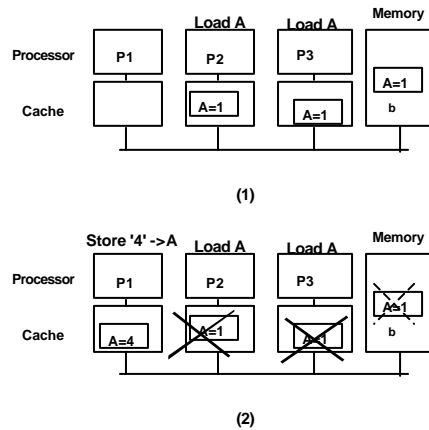


- Threads are sharing process resources
- Cost of thread switching is lower than process switching

Page 6
© R.J Chevrance

Microprocessors and SMP

- Microprocessors must integrate specific features to operate in an SMP environment
- Cache Coherence - Illustration

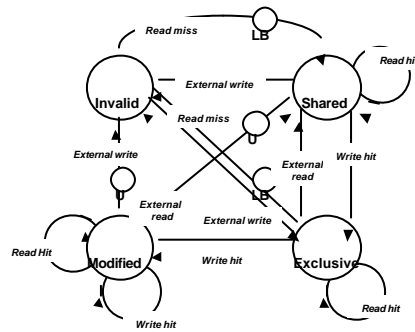


Page 7

© R.J Chevence

Microprocessors and SMP(2)

- Cache Coherence Protocol (e.g. MESI)



U indicates an update of the memory
LB indicates loading a block into the cache

- Memory Consistency
- Synchronization Mechanisms

Page 8

© R.J Chevence

Operating Systems and SMP

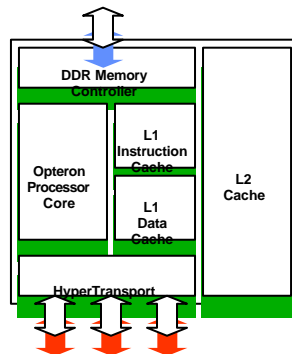
- Operating systems must be designed for SMP (e.g. Windows NT) or adapted to SMP (e.g. Unix)
- e.g. transforming an OS for SMP:
 - MP Safe Operation then MP efficient
 - Transforming kernel processes into threads
 - Pre-emptive kernel
 - Locking to prevent simultaneous access to shared resources (e.g. data structures)
 - Granularity, number of locks, duration of locks
 - Dead locks
 - Allocating processes/threads to processors (affinity)
 -
- Need constant tuning to adapt to changing characteristics such as the number of processors,....

Page 9

© R.J Cheavance

SMP with moderate number of processors

- By 2003, we can assume that up to 8 is a moderate number of processors (no need to use sophisticated technologies). This limit is growing with time as microprocessor vendors integrate more and more system capability into their silicon.
- Internal Structure of the AMD Opteron Processor (according to AMD)

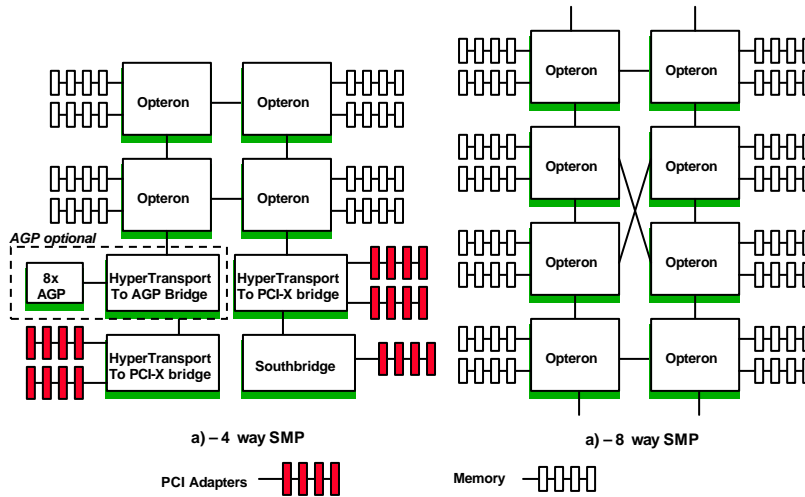


Page 10

© R.J Cheavance

SMP with moderate number of processors(2)

■ 4 and 8 processor HyperTransport-based SMP Systems (according to AMD)

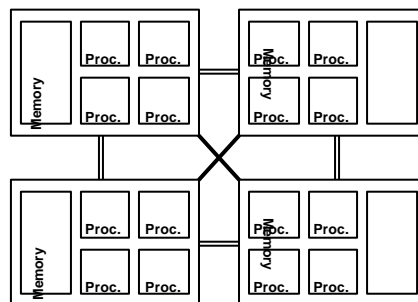


Page 11

© R.J Chevence

SMP with moderate number of processors(3)

■ Overview of IBM's X-Architecture (IBM)

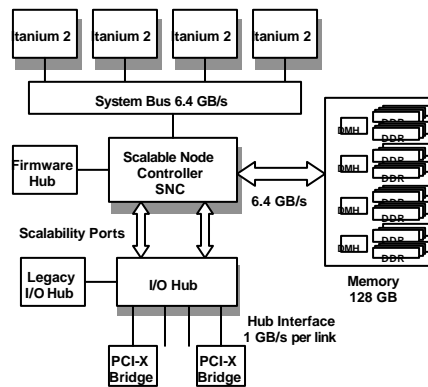


Page 12

© R.J Chevence

SMP with moderate number of processors(4)

■ Architecture of a 4-Way Itanium-2 Based SMP Server, the SR870NB4 (Source INTEL)

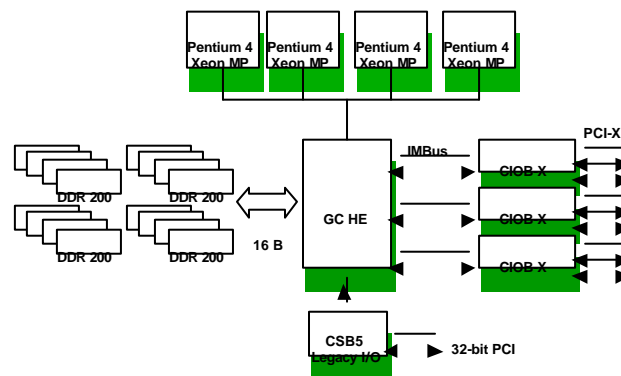


Page 13

© R.J Cheavance

SMP with moderate number of processors(5)

■ Architecture of a 4-Way Xeon MP-based Server (according to ServerWorks)

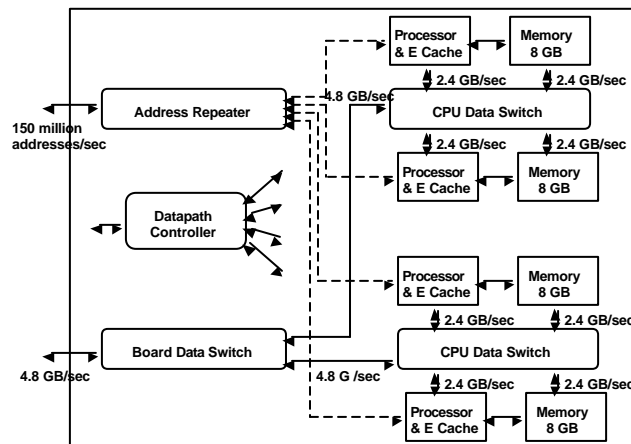


Page 14

© R.J Cheavance

SMP with moderate number of processors(6)

■ Crossbar-based Architectures: Sun Fire 3800 Architecture (Source: Sun)



Page 15

© R.J Cheavance

SMP with Many Processors ≥ 16 processors

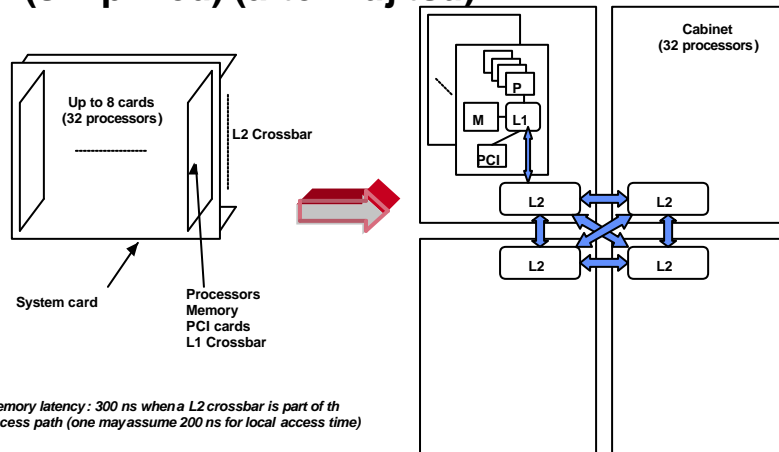
- Because of the physical and electrical constraints, larger SMP systems must use a hierarchy of interconnect mechanisms; in a recursive manner, a large SMP tends to be a collection of smaller systems, each of which is itself an SMP
- Such a hierarchy means that memory access times differ - when a processor accesses its own memory, it will see a much shorter access time than when it accesses memory in a distant module. This non-uniformity of memory access times can have a major effect on system behavior and on the software running on it
- The degree of uniformity allows us to distinguish two classes of SMP systems.
 - Systems in which access to any memory takes an identical amount of time, or very nearly identical are called (respectively) UMA - for Uniform Memory Access - or nearly UMA .
 - Systems in which there is a noticeable disparity in access times - a factor greater than 2 - are referred to as CC-NUMA, for Cache-Coherent Non-Uniform Memory Access
- This classification is purely empirical, and its practical significance is that in a distinctly NUMA system, software must - for maximum efficiency - take account of the varying access times. The amount of work needed for this adaptation tends to increase with the NUMA ratio.

Page 16

© R.J Cheavance

Nearly UMA Systems

■ Fujitsu Prime Power System architecture (simplified) (after Fujitsu)



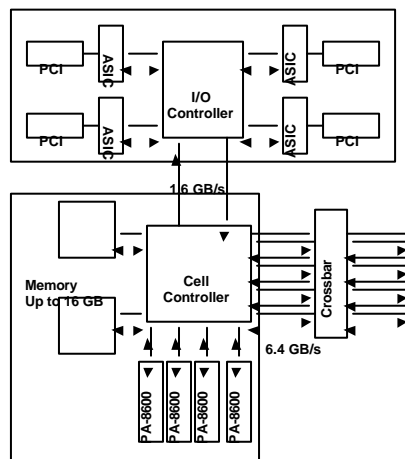
Page 17

© R.J Chevence

Nearly UMA Systems(2)

■ HP Superdome

□ HP Superdome Cell (after HP)

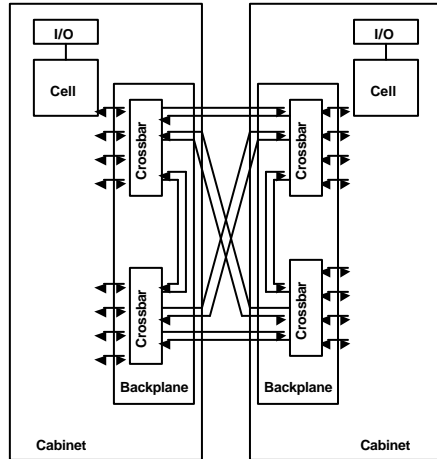


Page 18

© R.J Chevence

Nearly UMA Systems(3)

■ HP Superdome Architecture (after HP)



Memory latencies for a fully-loaded system are said to be:

- within-cell: 260 ns;
- on the same crossbar: 320 to 350 ns;
- within the same cabinet: 395 ns;
- between cabinets: 415 ns.

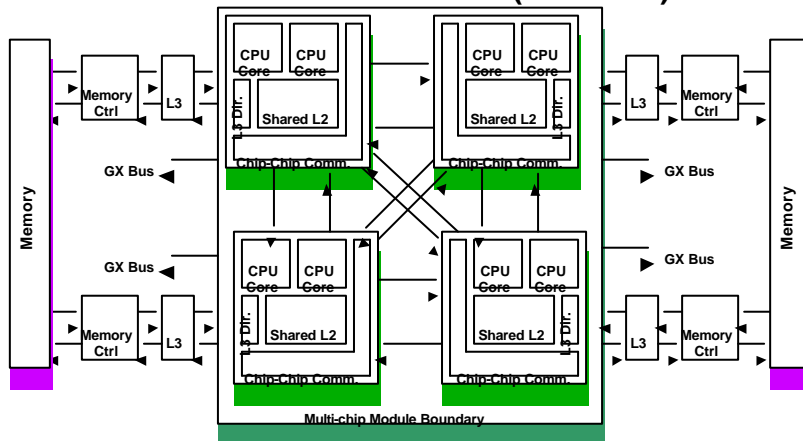
Page 19

© R.J Chevanca

Nearly UMA Systems(4)

■ IBM pSeries 690

□ Structure of Power4 MCM (after IBM)

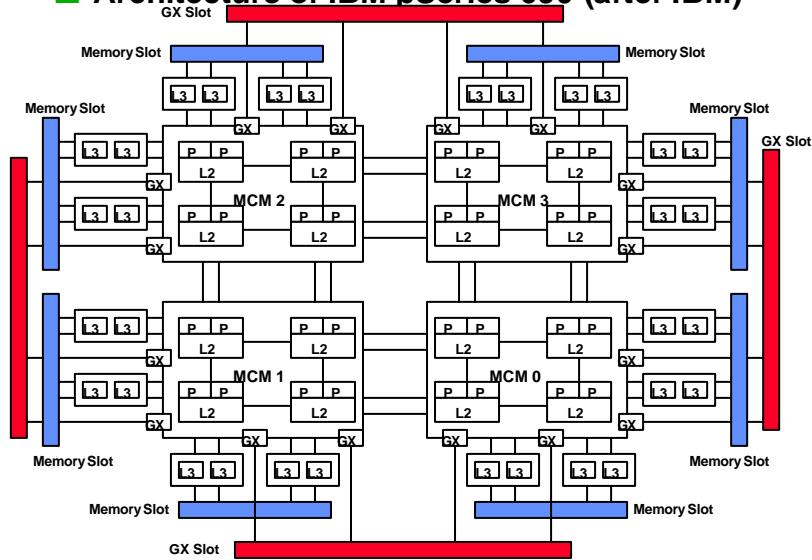


Page 20

© R.J Chevanca

Nearly UMA Systems(5)

Architecture of IBM pSeries 690 (after IBM)

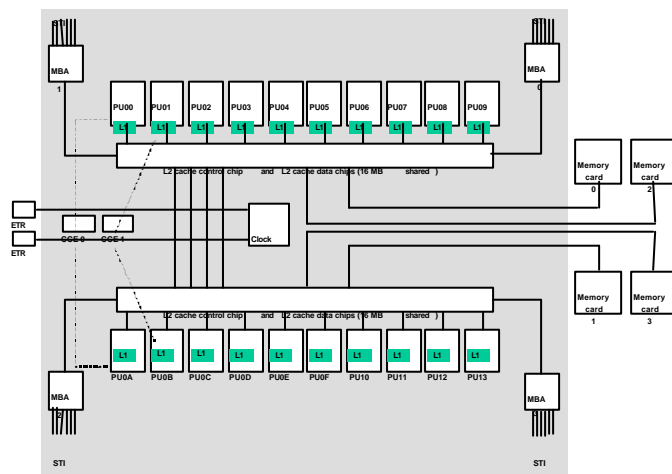


Page 21

© R.J Chevence

Nearly UMA Systems(6)

z900 Hardware Architecture (after IBM)

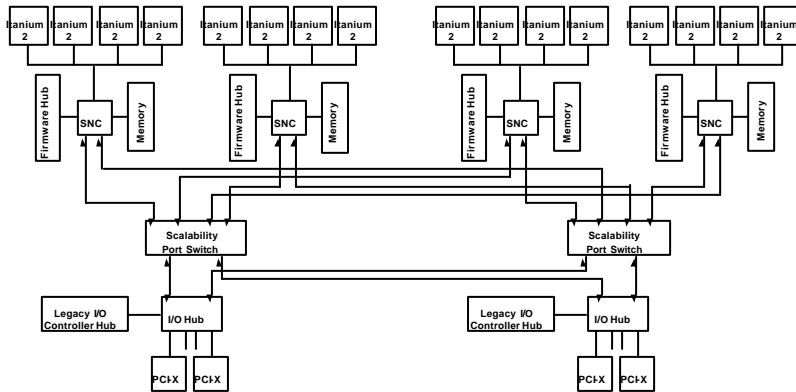


Page 22

© R.J Chevence

Nearly UMA Systems(7)

■ Architecture of a system with 16 Itanium 2 processors (after INTEL)



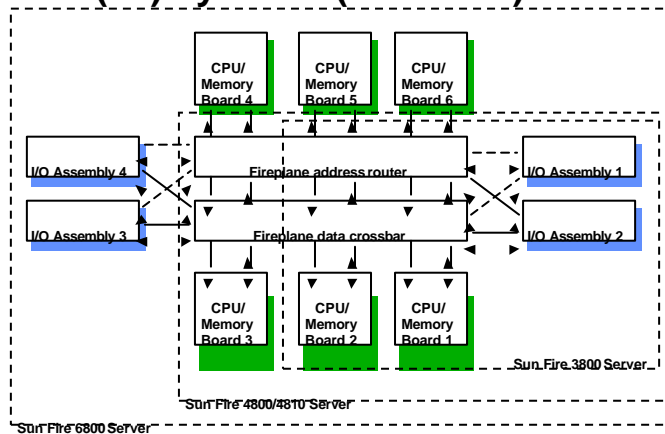
The system is based on 4-processor building blocks, using - here - the Scalability Port Switch and its Scalability Ports. These provide both connection and coherence, and offer a reasonable NUMA factor - Intel indicates a value of 2.2 for this system

Page 23

© R.J Cheavance

Nearly UMA Systems(8)

■ Structure Sun's 3800(8), 4800(12) and 6800(24) Systems (after Sun)



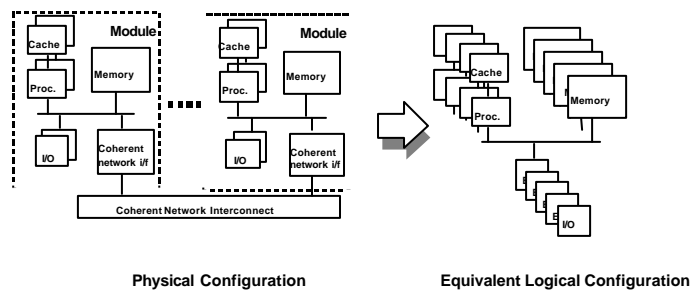
Page 24

Local memory access time: 180 ns, access to memory on another card takes 240 ns

© R.J Cheavance

CC-NUMA Architecture

- Early CC-NUMA systems were based upon 4-way building blocks (Pentium-based) and used derivatives of the SCI (Scalable Coherent Interface). Early systems were characterized by poor performance
- Principles of CC-NUMA Architecture



Page 25

© R.J Cheavance

CC-NUMA Architecture(2)

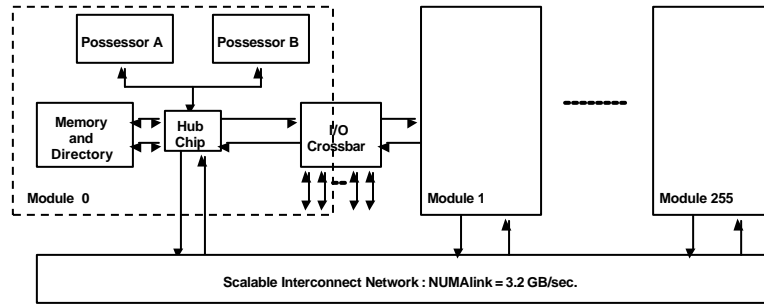
- CC-NUMA performance is very sensitive to locality of information. Large NUMA factor (e.g. above 5) requires extensive software optimizations
- Some examples of possible optimizations when NUMA factor is large:
 - Affinity scheduling: arranging that a process, once suspended, is rescheduled on the same processor on which it last executed in the expectation that the caches contain useful state from the earlier execution
 - When allocating memory, prioritize local allocation (i.e., allocation within the module on which the program is executing) above distant allocation
 - When a process has had pages paged out, allocate memory for the pages when paged back in on the same module as the process is executing

Page 26

© R.J Cheavance

CC-NUMA Architecture(3)

■ General Architecture of SGI Origin 3000 System (after SGI)

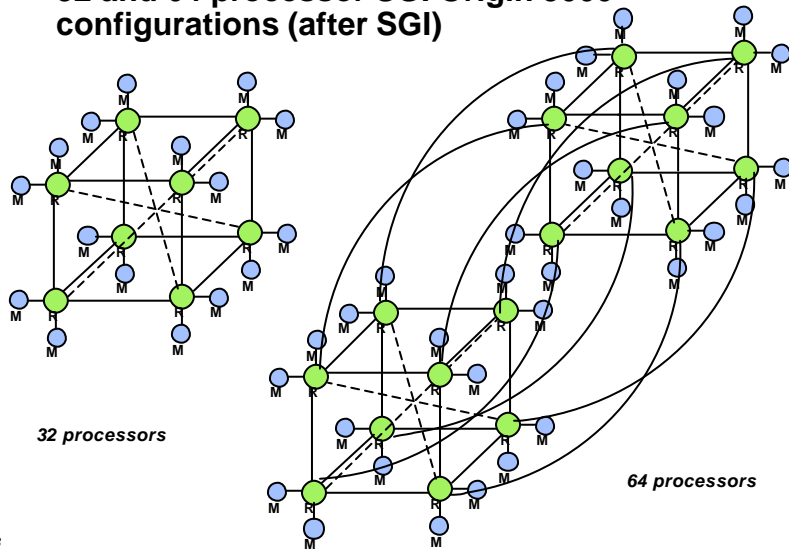


Page 27

© R.J Cheavance

CC-NUMA Architecture(4)

■ 32 and 64 processor SGI Origin 3000 configurations (after SGI)

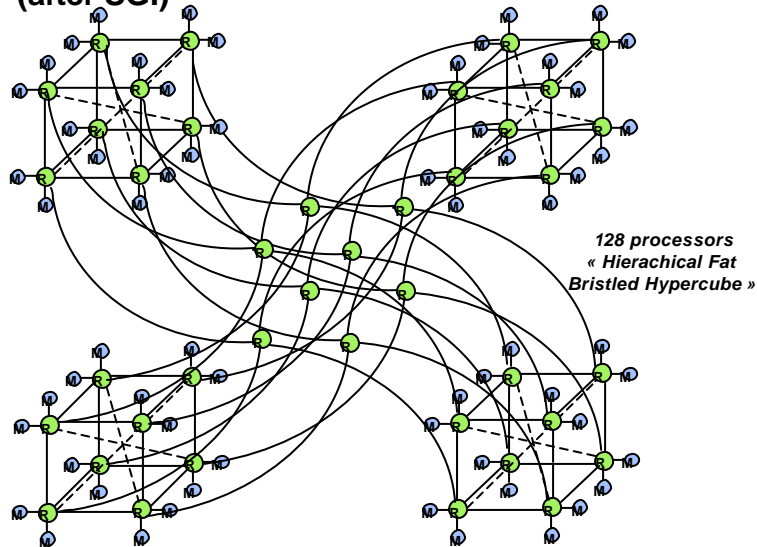


Page 28

© R.J Cheavance

CC-NUMA Architecture(5)

■ SGI Origin 3000 - 128-processor Configuration (after SGI)



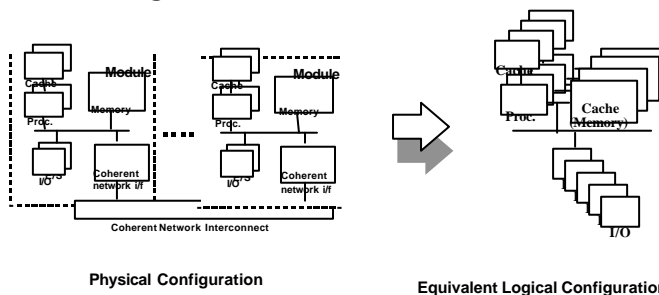
Page 29

© R.J Cheavance

COMA Architecture

- **COMA Architecture (Cache Only Memory Architecture)** aimed at large SMP: data doesn't have a fixed place in memory (that is, has no fixed address in the memory of a module), and the memory of each module acts as a cache. Data is moved, by hardware, to where it is being accessed, and data that is shared for reading may therefore be resident in several modules at any one time

- **Schematic diagram of a COMA architecture**



Page 30

© R.J Cheavance

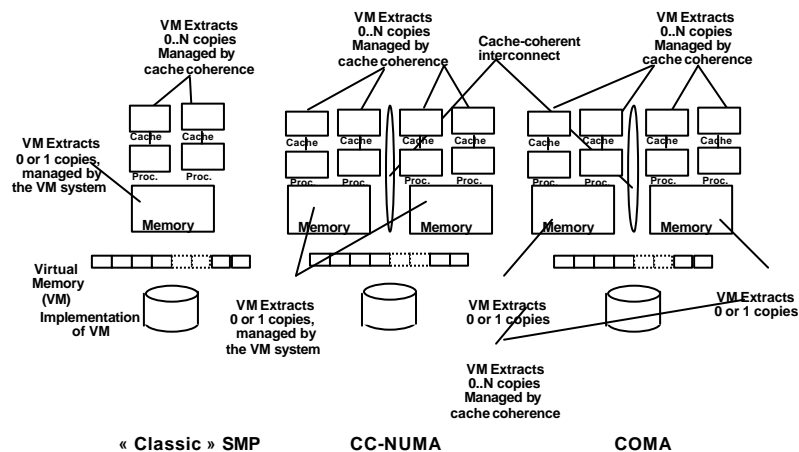
COMA Architecture(2)

- Only one Company (KSR now disappeared) has introduced a COMA system to the market place
- Advantages:
 - because memory is managed as a cache, data is always local to the module using it
 - many copies of data used read-only may exist simultaneously
- Disadvantages:
 - the hardware is more complex
 - directory sizes increase, as does the time necessary to make allocate data
- As originally described, COMA required a specific microprocessor architecture in the area of memory management (e.g. KSR) and so a very large investment
- S-COMA (Simple COMA), a project at Stanford University, is using standard microprocessors

Page 31

© R.J Chevalance

SMP, CC-NUMA, COMA: a Summary



- The traditional UMA SMP architecture provides the best characteristics as regards memory access time and the simplicity of the required cache coherence protocols
- CC-NUMA's data uniqueness property (0 or 1 copies of an object in memory at any time) simplifies the coherence protocol at the expense of increased access time to data in other modules
- COMA's ability to replicate data copies improves effective access time but results in a greater complexity in the needed cache coherence protocol

Page 32

© R.J Chevalance

Performance Improvement Provided by SMP

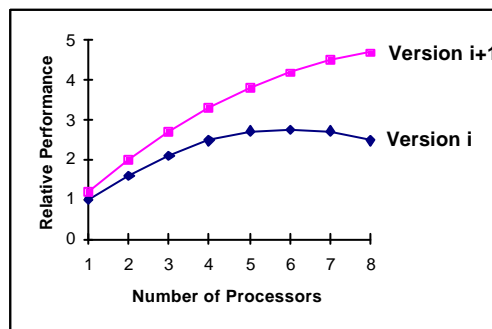
- **A number of factors prevents an N-processor SMP having N times the performance of a uniprocessor e.g.**
 - Contention for the bandwidth of the processor-memory interconnect increases with the number of processors
 - Contention for shared memory increases with the number of processors
 - Maintaining cache coherence means that cache line invalidations, and movement of data from cache to cache, must occur
 - Reduction of cache hit rate as a result of more context switches
 - An increase in the number of instructions which must be executed to implement a given function because of the existence of locks and the contention for them
- **Very few data publicly available about scalability of SMP**

Page 33

© R.J Chevance

Performance Improvement Provided by SMP(2)

- **Evolution of the behavior of a system multiprocessor (transactional application and DBMS) according to the successive versions of software showing the importance of improvements in the OS and the DBMS (the hardware being unchanged)**



Page 34

© R.J Chevance

Advantages and disadvantages of SMP architecture

Advantages	Disadvantages
. Scalability of the system at a moderate cost.	. Existence of at least one single point of failure - the operating system
. Simple performance increase by adding a card or a processor module	. Maintenance or update activities on the hardware and the OS generally require the whole system to be stopped
. Multiprocessor effectiveness - the system performance increases (within definite limits) as the number of processors grows	. Limitation of the number of processors because of access contention in the hardware (e.g., the bus) and software (operating system and DBMS's).
. Simplicity of the programming model	. Upgrade capability limited by rapid system obsolescence versus processor generations
. Application transparency - single processor applications continue to work (although only multi-threaded applications can benefit from the architecture)	. Complexity of the hardware.
. Availability: if one processor fails, the system may be restarted with fewer processors	. Adaptation and expensive tuning of the operating system.
. The ability to partition the system	. Necessary application adaptation to benefit from the performance available.

Page 35

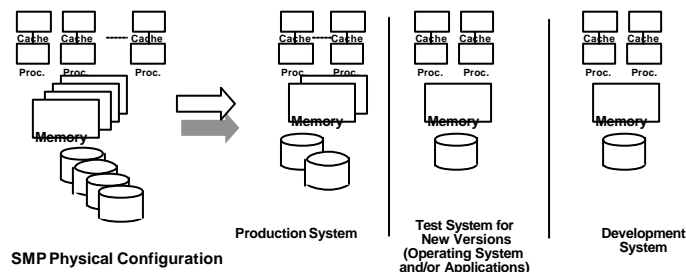
© R.J Chevanee

Partitioning

- **Partitioning divides a single physical SMP system into a number of independent logical SMPs, each logical SMP running its own OS. Examples of situations where partitioning is useful:**

- when consolidating several servers into a single system (for investment rationalization, for example), while maintaining the apparent independence of the servers;
- sharing the same systems between independent activities whose balance changes over time

- **Example of system partitioning:**



Page 36

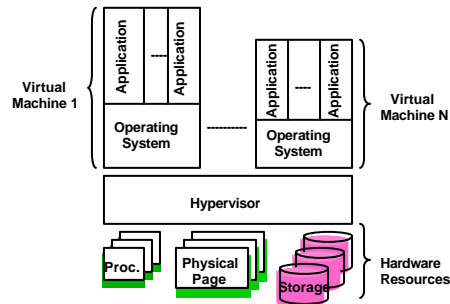
© R.J Chevanee

Partitioning(2)

■ Techniques of Partitioning

- Virtual Machines (introduced in the late 60's for mainframes and widely used, being now used for servers based upon standard technologies)

- Virtual Machine - Principles of Operation



- Each virtual machine has its own address space and Operating System (all can be different)
- The Hypervisor executes privileged mode instructions issued by guest OS onto real resources

Page 37

© R.J. Chevence

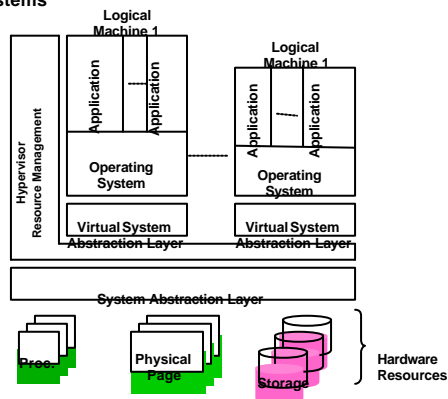
Partitioning(3)

■ Hardware Partitioning

- Consists of defining, for a given physical system configuration, a number of logical systems, each with its own allocation of hardware resources (processor, memory, controllers and peripherals)

■ Logical Partitioning

- Logical partitioning does not require dedication of the physical resources to logical systems; rather, the resources are managed by a special operating system which dynamically multiplexes the physical resources correctly between the logical systems



Page 38

© R.J. Chevence

Partitioning(4)

- **Dynamic partitioning is key for the flexibility of the operation (otherwise the whole system must be stopped, reconfigured and restarted). Virtual Machines and Logical Partitioning facilitate dynamic partitioning**
- **Workload Managers:**
 - **Complements an OS by ensuring that a given application (or set of applications) can be afforded a guaranteed slice of the actual physical systems resources (e.g, physical memory, processor time.)**
 - **A Resource Manager reserves sufficient resources to ensure that critical work may be performed as required, while allowing a system so support multiple concurrent applications. When the critical work does not use all pre-allocated resources, the unused resource slots may be used by other applications**

Page 39

© R.J Cheavance

Server Consolidation

- **The pressure on costs (TCO reduction), information managers are looking at reducing the number of servers deployed. Several approaches are proposed such as:**
 - **physical concentration were several servers are regrouped in the same location possibly in a cluster configuration (offering high availability if needed);**
 - **concentration were all applications distributed on several servers are supported on a single system (i.e. in a way similar to the “traditional” mainframe) with technologies such as:**
 - **Virtual machine approach**
 - **Workload management were a single operating system supports simultaneously several applications. Resource sharing between applications is being managed according to a set of stated rules.**
 - **data centric were data is supported by a single (high availability) system, offering to application servers the data service (application servers does not support data) ;**
 - **application integration were various applications and associated data are concentrated on a smaller set of servers in order to facilitate the cooperation between applications.**
- **The choice of a solution depends heavily on the characteristics of the existing configuration and the set of constraints. Obviously, reducing the number of servers contributes to reduce the TCO. Both cost and business continuity issues may prevent to go further than physical concentration.**

Page 40

© R.J Cheavance