Multicore Operating Systems

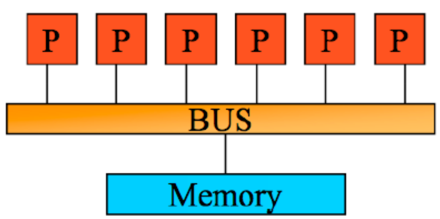
Minsoo Ryu

Department of Computer Science and Engineering
Hanyang University

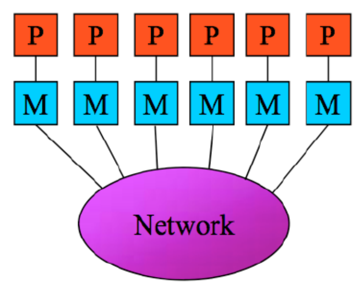
Outline

Considerations for Multiprocessors

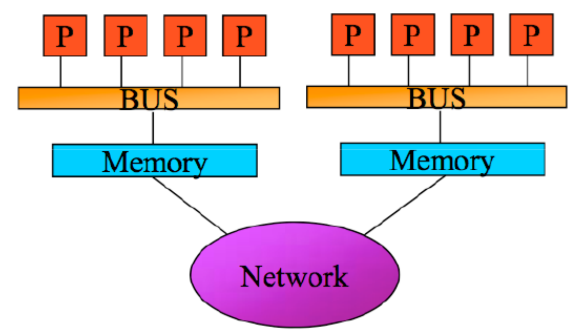
- Diversity in HW organization
  - Shared memory vs. distributed memory
  - Bus-based vs. network-based
  - Homogeneous vs. heterogeneous



Shared memory & bus-based (UMA)



Distributed memory & network-based (NUMA)



Clustered & hybrid (NUMA)

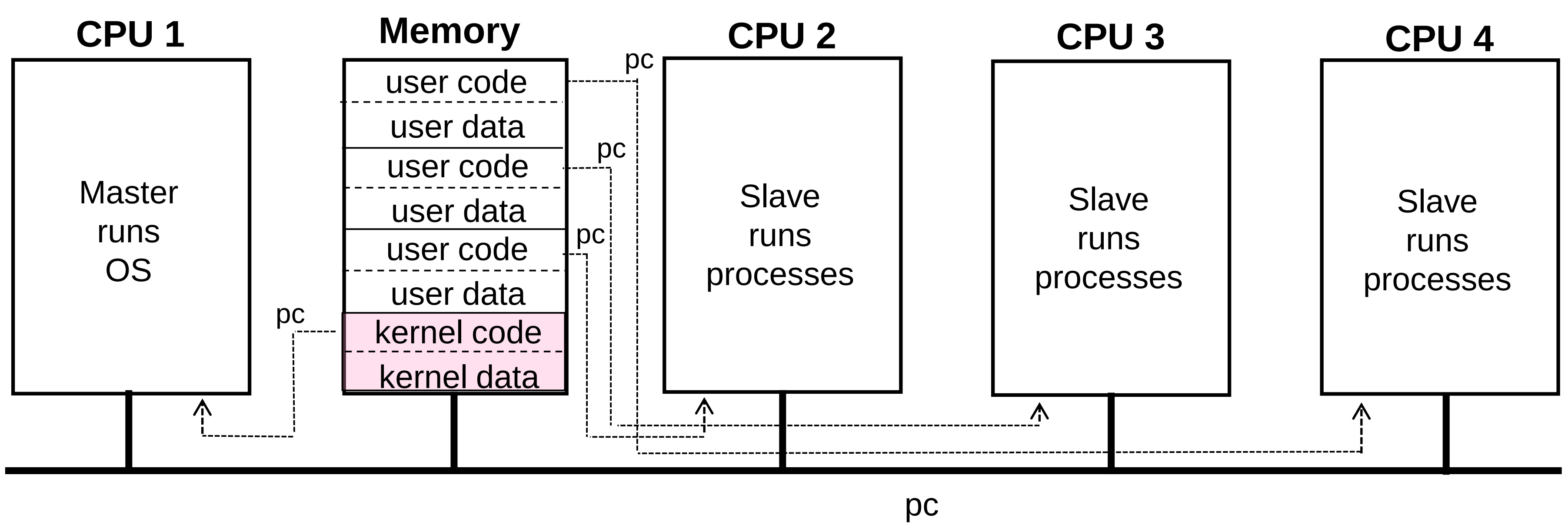
- Interactions among processors
  - Memory and cache sharing
  - Inter-processor communications

Classification of Multiprocessor Kernels

- For small multiprocessors
  - Master-slave organization
  - Independent kernels
  - SMP (Symmetric Multiprocessing) kernel
- For large multiprocessors (more than tens of cores)
  - Distributed kernels
  - Factored kernels
  - Clustered organization

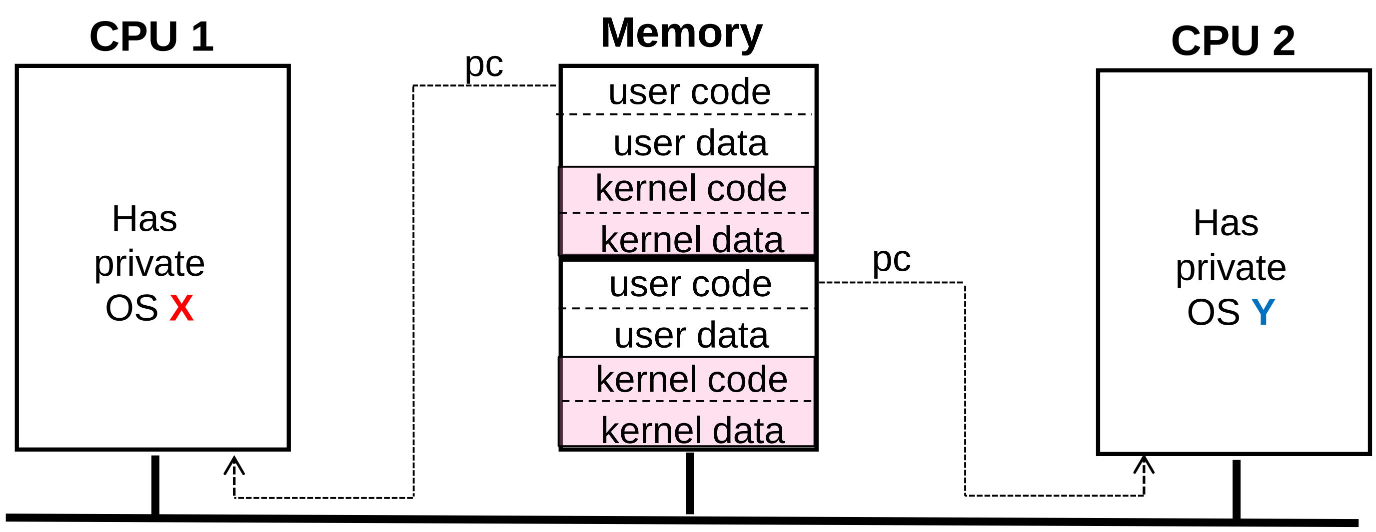
Master-Slave Organization

- One copy of the OS on a master CPU
  - All system calls are redirected to the master CPU
  - All the other CPUs run user processes
- Problem
  - The master CPU can be a bottleneck



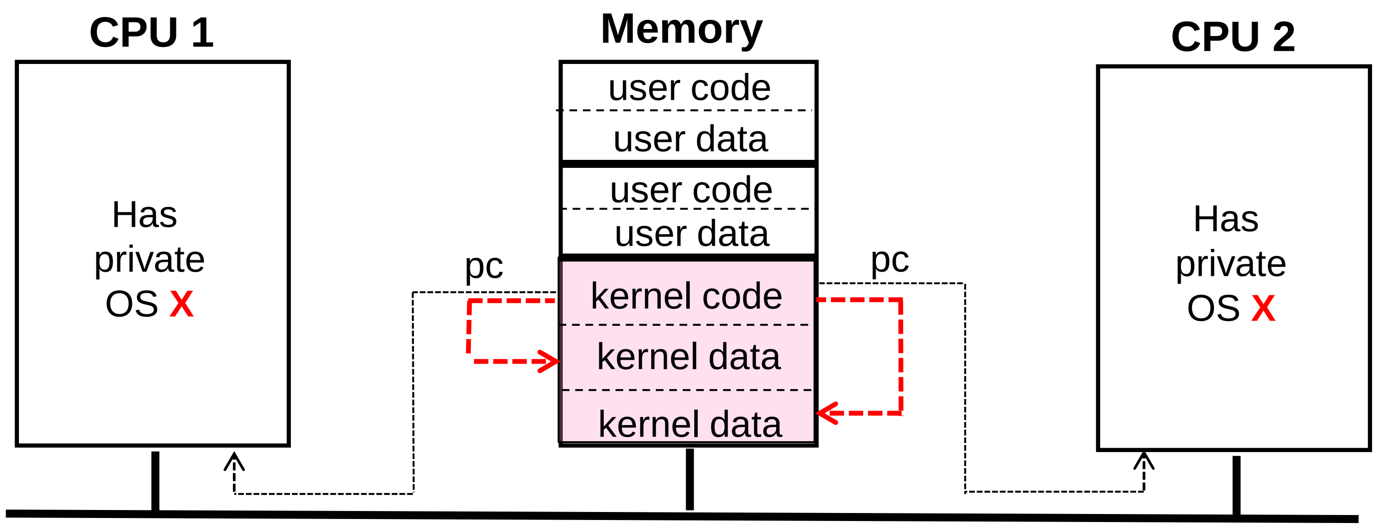
Independent Kernels

- Each CPU has its own private OS
  - Memory is partitioned and assigned to each OS as private memory
  - The *n* CPUs operate as *n* independent computers



Independent Kernel Instances

- Each CPU has a different instance of the same OS
  - All the CPUs share the OS code and make private copies of only the data



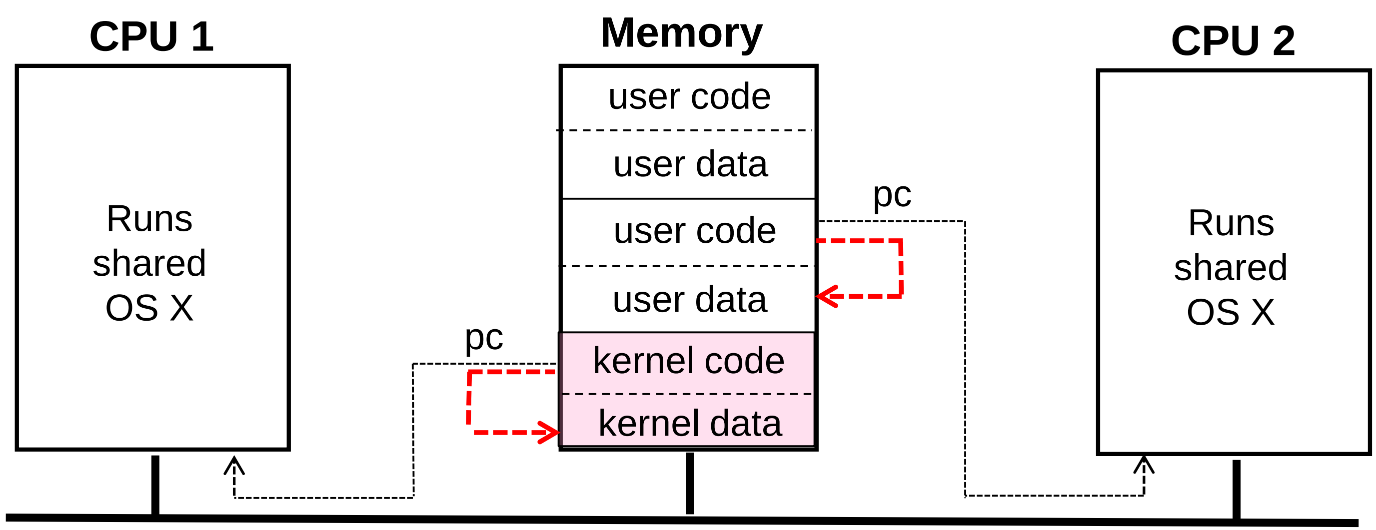
Advantages and Disadvantages

- **Advantages**
  - **Better than *n* separate computers since it allows all the machines to share a set of hardware resources**
  - **A single kernel crash does not affect other kernels**

- **Disadvantages**
  - **Applications need to written in a distributed fashion**
    - Cannot exploit the tightly coupled hardware architecture
  - **There is no sharing of threads and each OS schedules their threads by itself**
  - **Using shared resources may result in inconsistent results**
    - e.g.) A certain block is present and dirty in multiple buffer caches at the same time

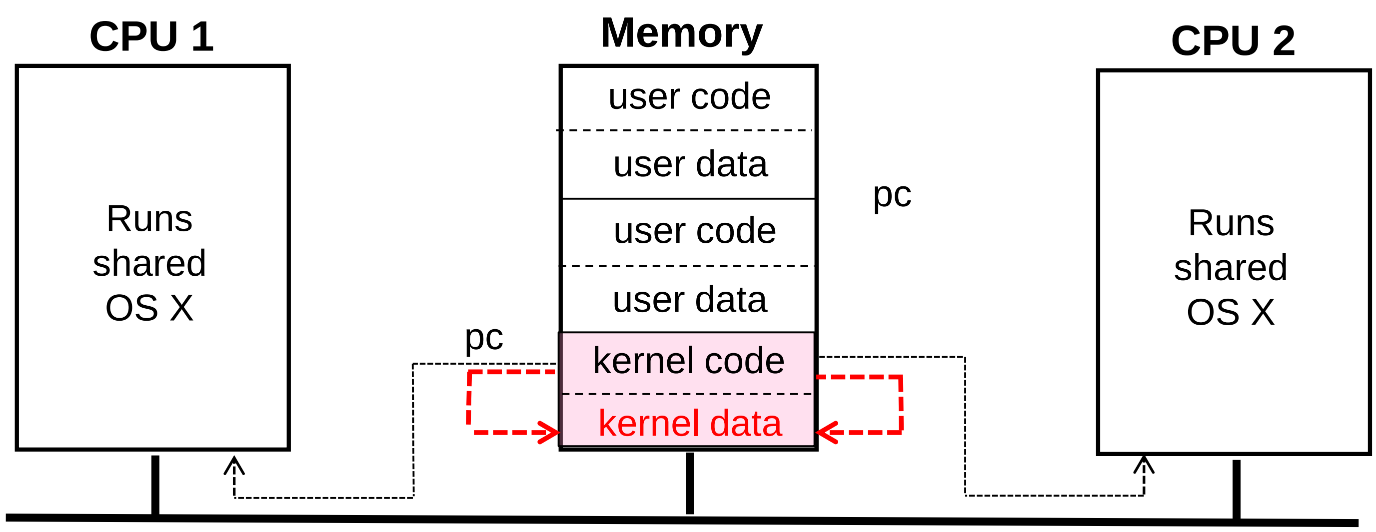
SMP (Symmetric Multiprocessing) Kernels

- One copy of the OS in memory and any CPU can run it
  - When a system call is made, the CPU on which the system call was made traps to the kernel and processes the system call



Data Races

- **However, there can be data races**
  - **Two CPUs can execute simultaneously in the kernel**
  - **They may access the same kernel data simultaneously**

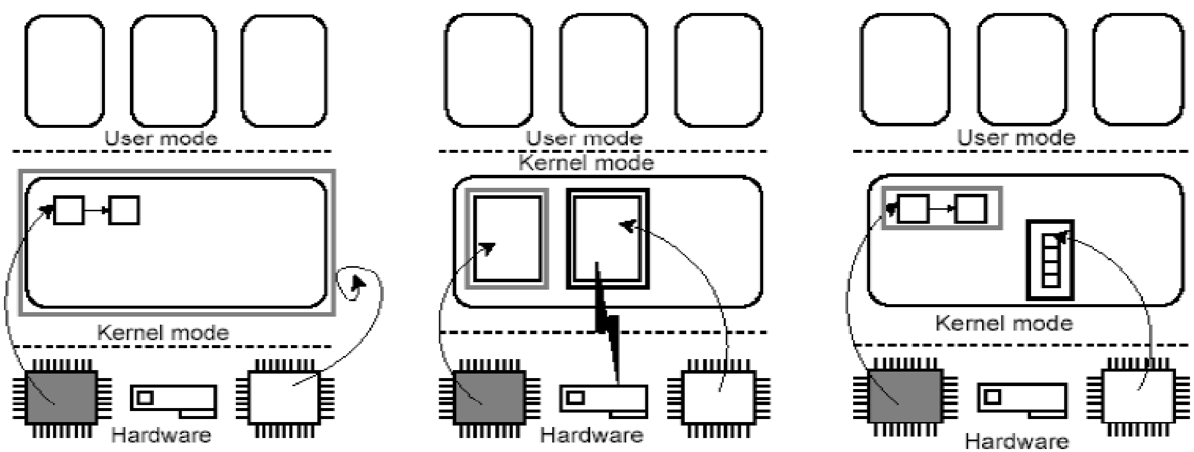


Giant Lock

- Simple solution
  - Associate a mutex (giant lock) with the OS to make the whole system one big critical region
  - When a CPU wants to run OS, it must acquire the mutex
- This solution is not scalable
  - If 10% of all run time is spent inside the OS, then 10 CPUs will pretty much saturate the whole system

Coarse- and Fine-Grained Locks

- **Better solution**
  - **Split the OS into independent critical regions**
    - One CPU can run the scheduler while another CPU is handling a system call
    - Note: great care must be taken to avoid deadlocks



(a) Giant lock (b) Coarse-grained (c) Fine-grained

SMP kernels are in widespread use: Unix, Linux, MS Windows, Apple OS X, iOS, …

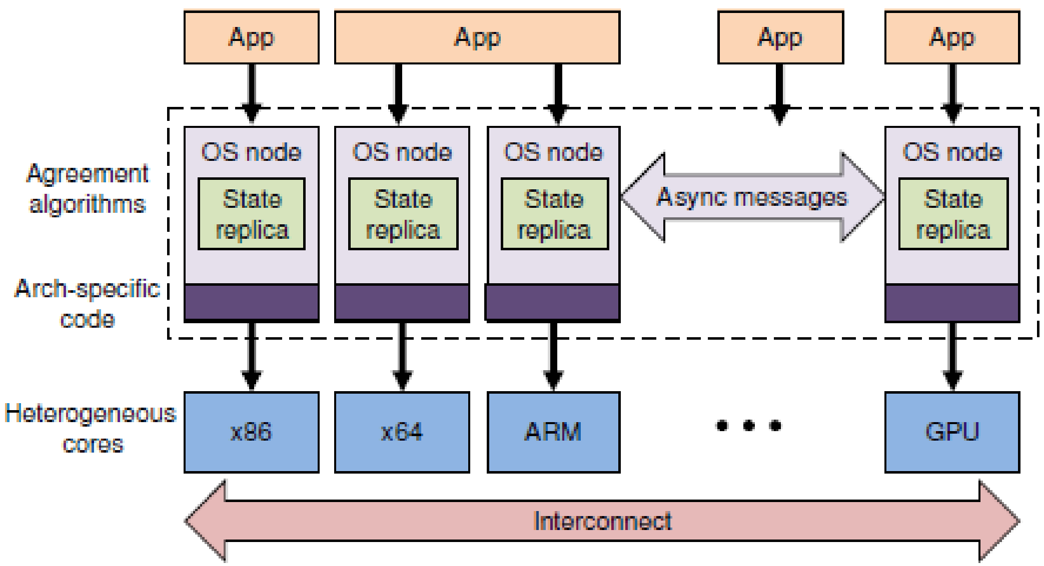
Advantages and Limitations of SMP kernels

- **Advantages: "shared everything structure"**
  - **All OS services and abstractions are shared**
    - e.g.) abstraction of a single flat, global address space
  - **All HW resources are shared**

- **Limitations**
  - **SMP kernels require restricted (SMP) HW architectures**
    - Identical instruction sets across processors
    - Shared memory structure
  - **SMP kernels do not scale well**
    - Lock contention cost
    - Reliance on shared memory

Distributed Kernels: The Multikernel Model

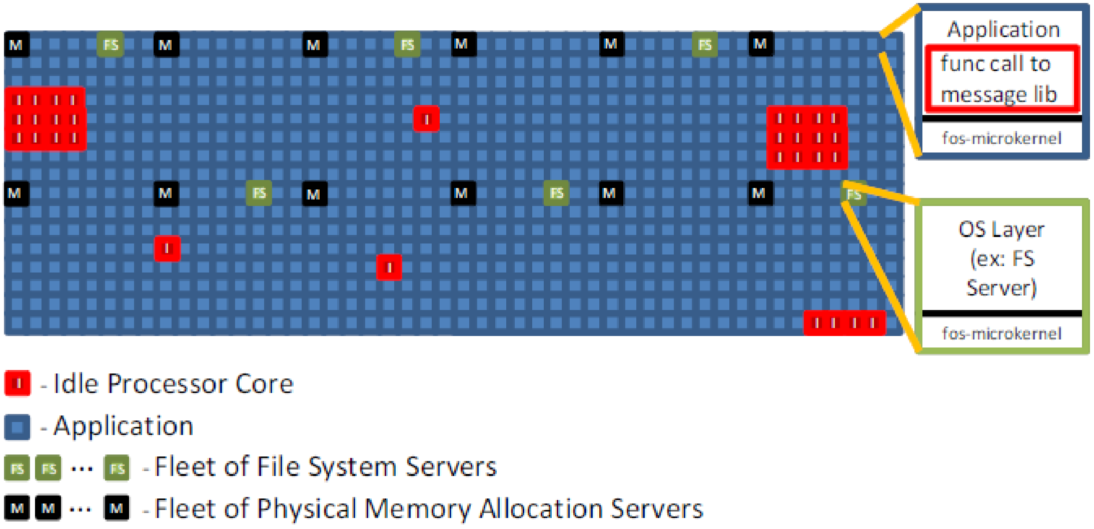
- Use ideas from distributed systems
  - Treats the machine as a network of independent cores
  - Replicates OS states across cores that communicate using messages and share no memory



Microsoft Research and ETH Zurich, ACM SOSP, 2009

Factored Kernels: FOS

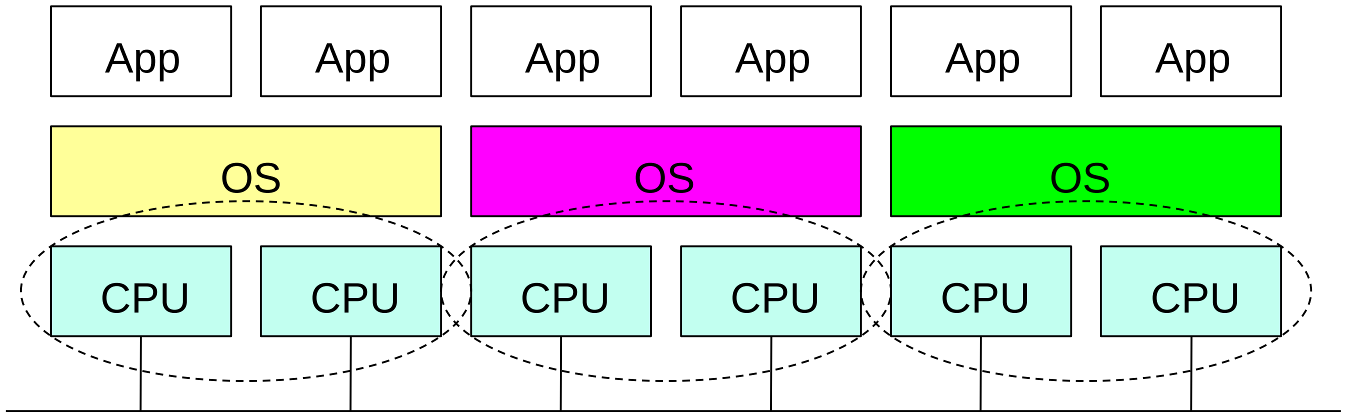
- A new operating system targeting manycore systems
  - Each operating system service is factored into a set of communicating servers
    - Space sharing rather than time sharing
  - Relies on message passing rather shared memory



MIT, ACM OSR, 2009

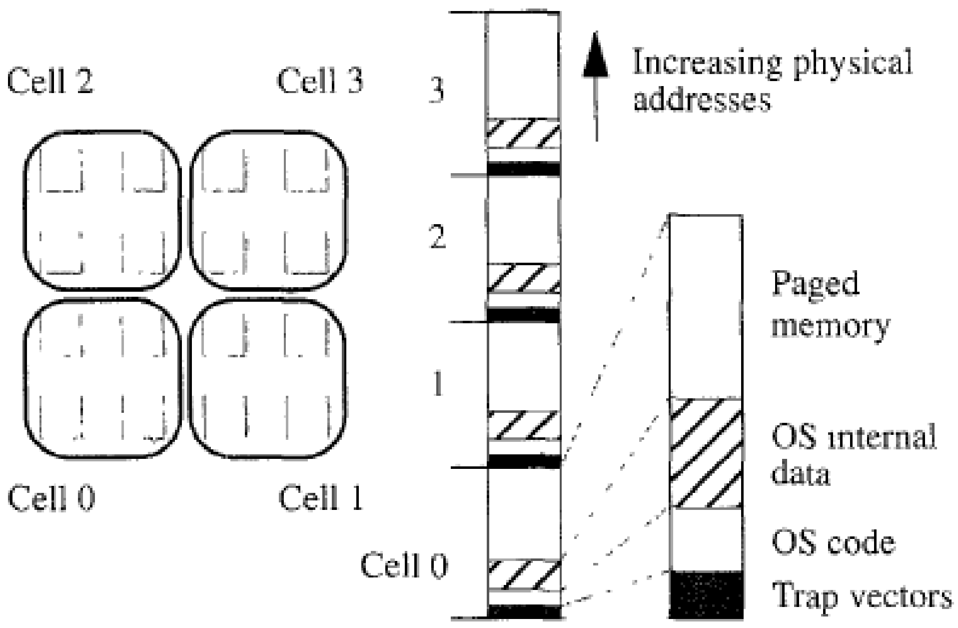
Clustered Organization

- Key idea
  - Processors are grouped into clusters
  - Each cluster runs an independent OS kernel
  - Clusters are the unit of scalability



Hive (Stanford, ACM SOSP'95)

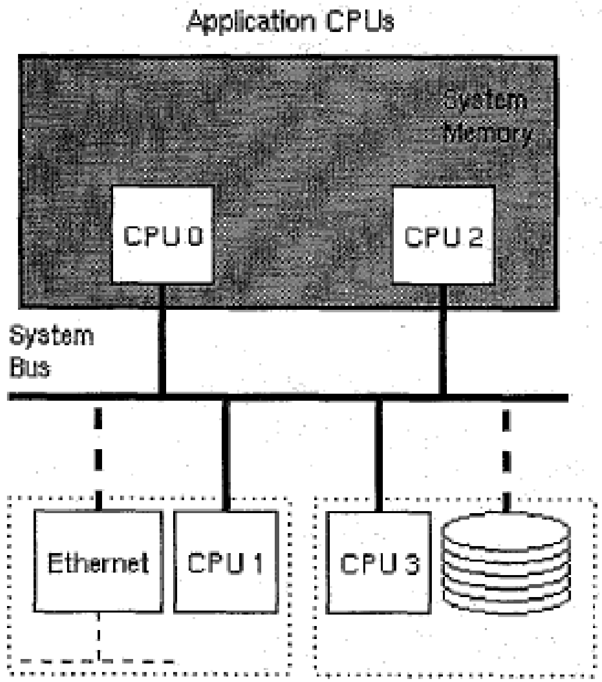
- The original purpose of Hive was to achieve reliability and fault containment
- Each cell runs an SMP kernel

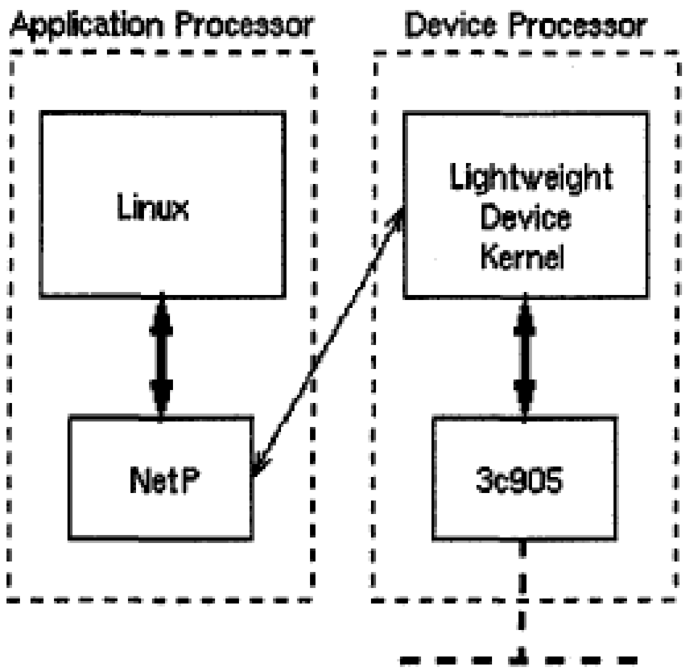


(a) Physical view (b) Memory layout

AsyMOS: Asymmetric Independent Kernels

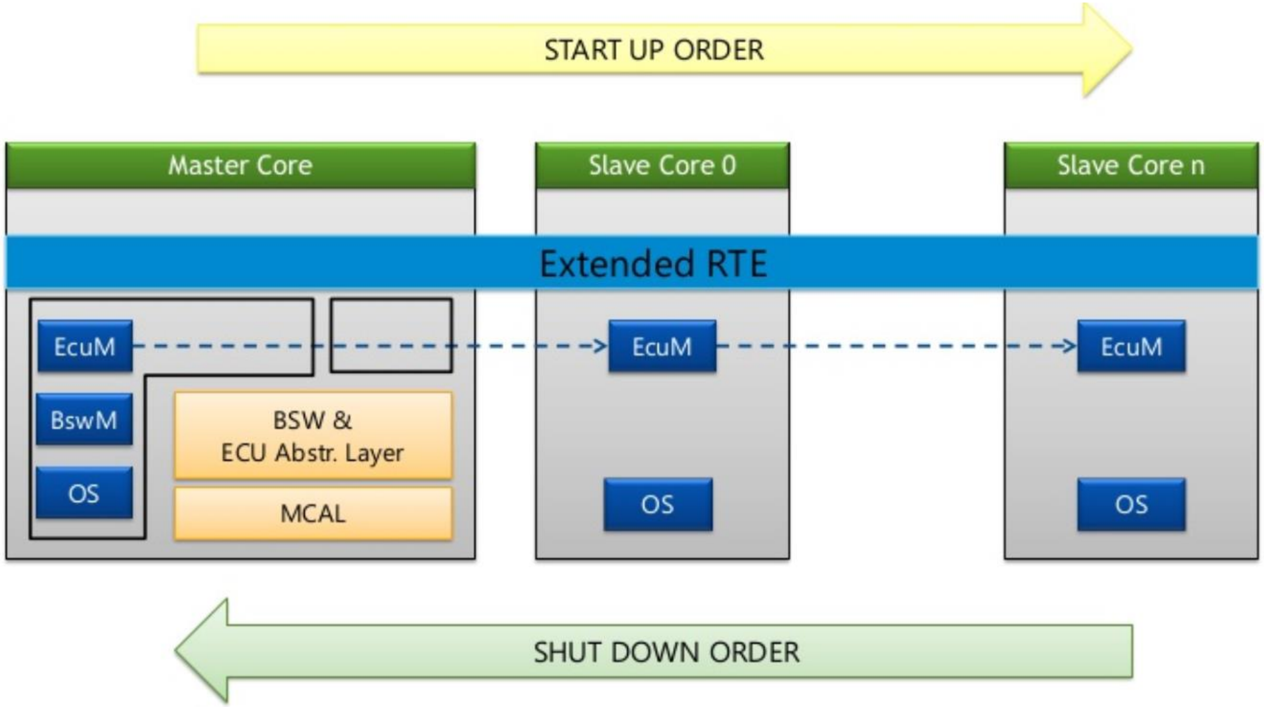
- Two types of kernel
  - A native OS kernel running on application processors
  - LDK (lightweight device kernel) running on device processors





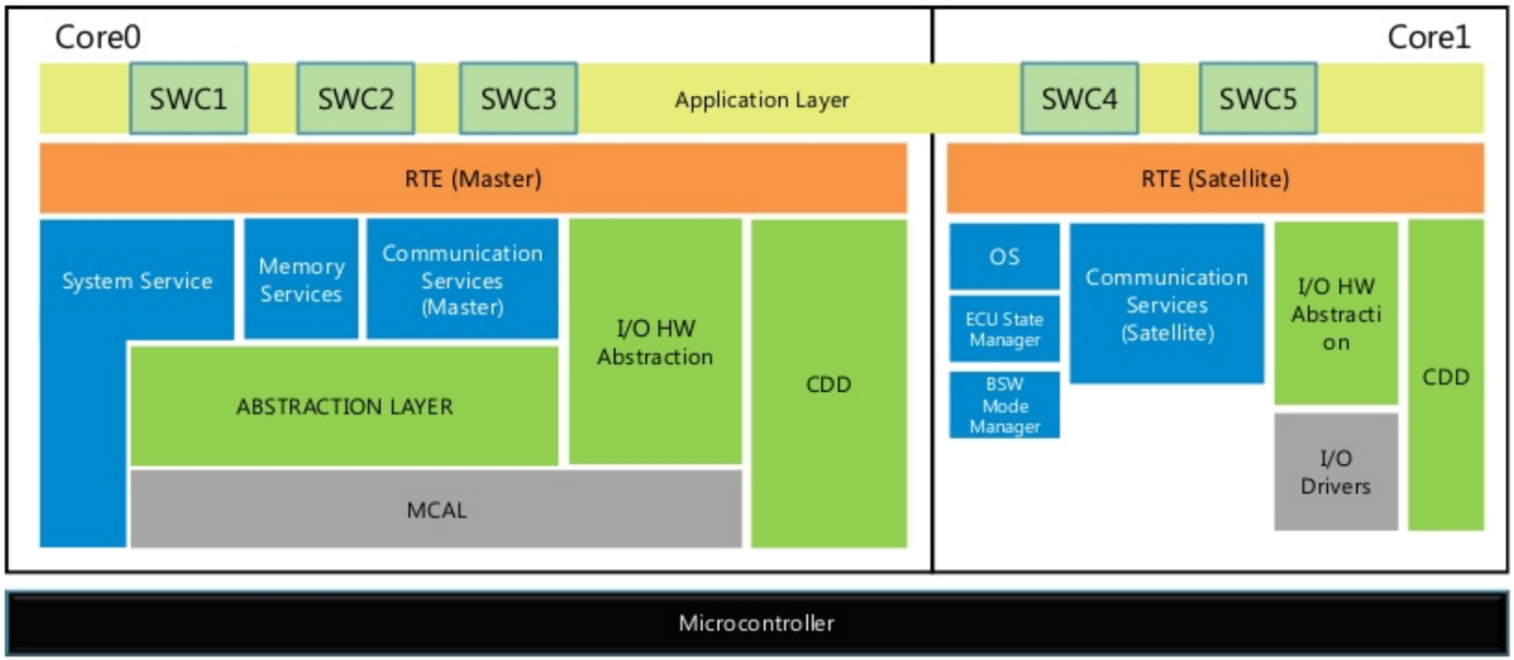
AUTOSAR Multicore: Master-Slave Concept

- Static assignment of OS-Applications to cores
- Master core contains complete BSW
- Master core controls start up and shut down



Master-Satellite Concept

- **The partitioning is implementation specific and the communication between master and satellite is not standardized**
  - **BSW can be partitioned (eg., a FlexRay cluster is on one core and a CAN cluster on a difference core)**



Spinlocks and IOC

- Spinlocks
  - protects a variable across core so that no two cores may access it at the same time
  - Is a busy waiting mechanism

- IOC (Inter OS-Application Communication)
  - Provides communication services which can be accessed by clients which need to communicate across cores and memory protection boundaries


Q&A