CE177

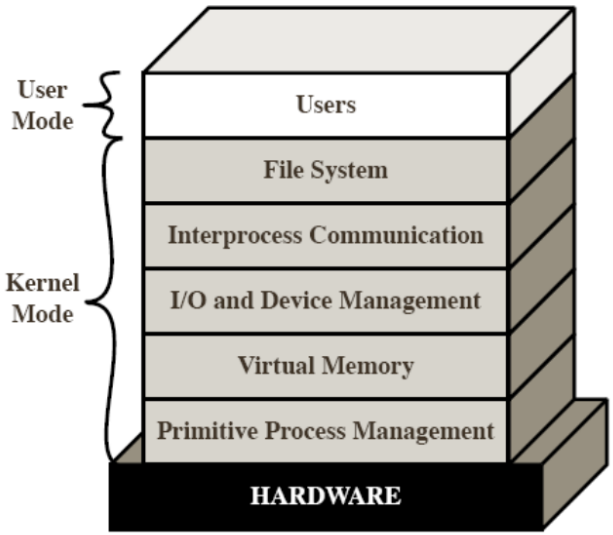
Advanced Operating Systems

Hamid Fadishei, Assistant Professor
Computer Engineerign Departemnt,
University of Bojnord
Fall 2018

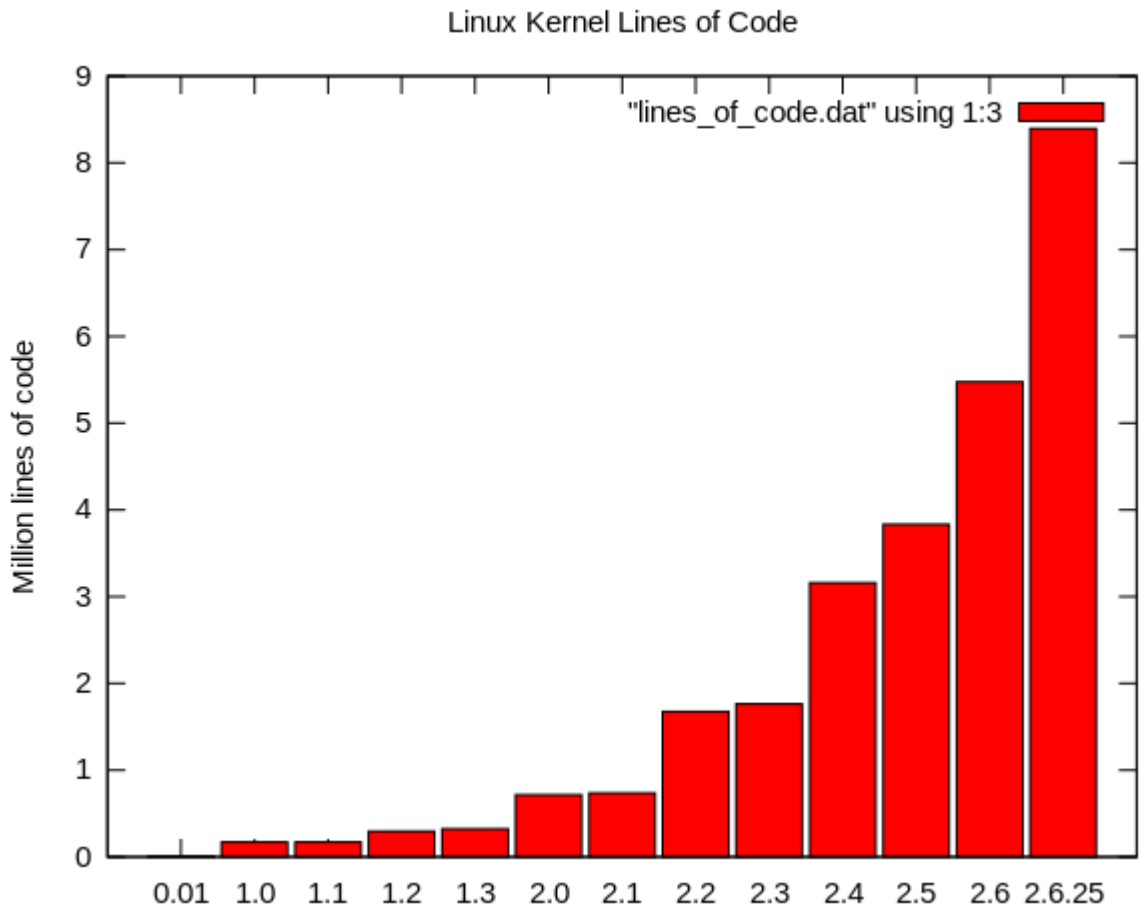
OS Architectures

Monolithic OS

- All OS services in periviledged mode
- Example: Linux



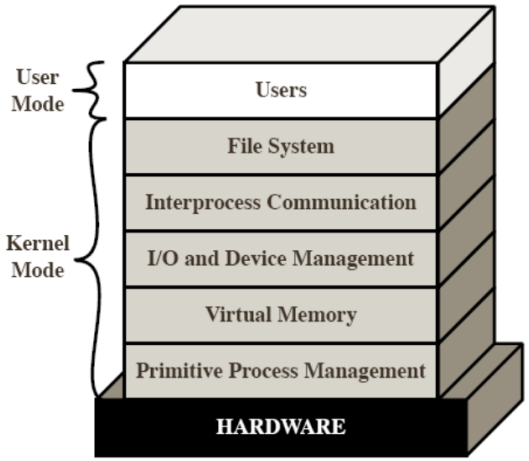
Linux kernel LOC

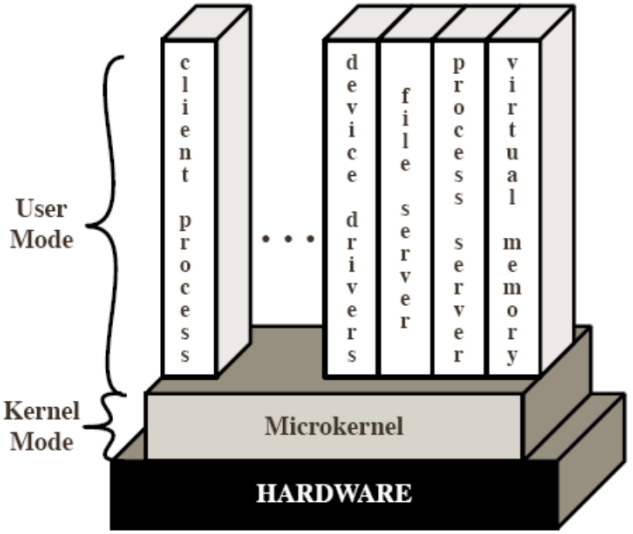


Microkernel

Microkernel

- Kernel mode: Minimal services
- The rest runs in user mode
- Philosophy: Not all OS services require privileged access
- Philosophy: Separate policy from mechanism
- Example: CMU Mach, MINIX



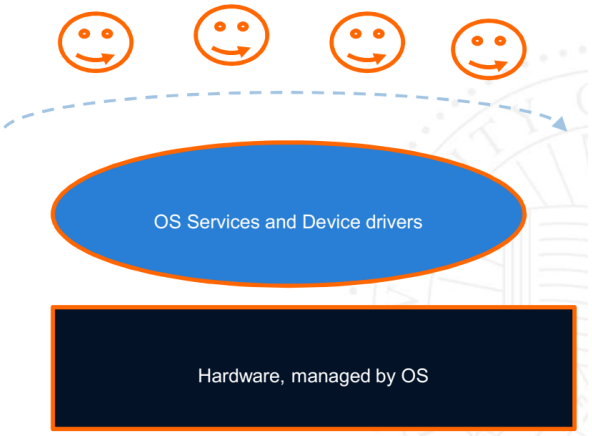


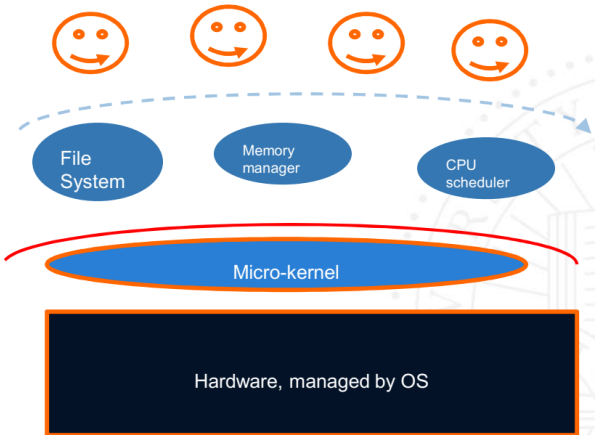
Microkernel benefits

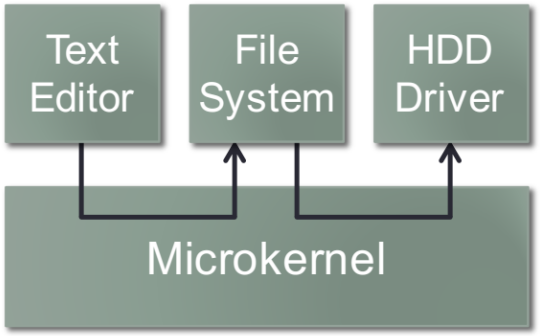
- Microkernels are…
  - Reliable
    - *Because small*
    - *Crash of a component does not necessarily crash the system*
  - Maintainable
  - Extensible
    - *It is easy to add features*

Microkernel drawback

- Performance overhead
  - Cost of crossing the lines
    - *IPC and Syscalls*



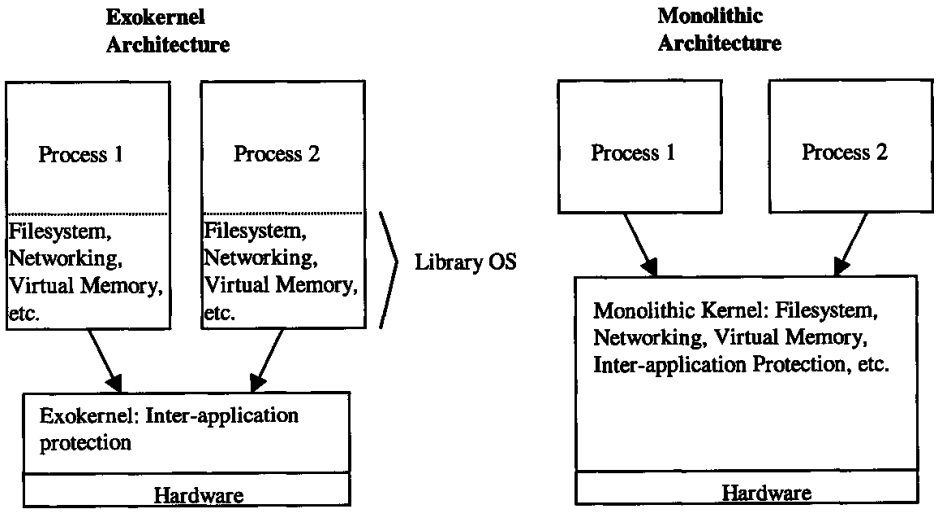




Exokernel/Unikernel

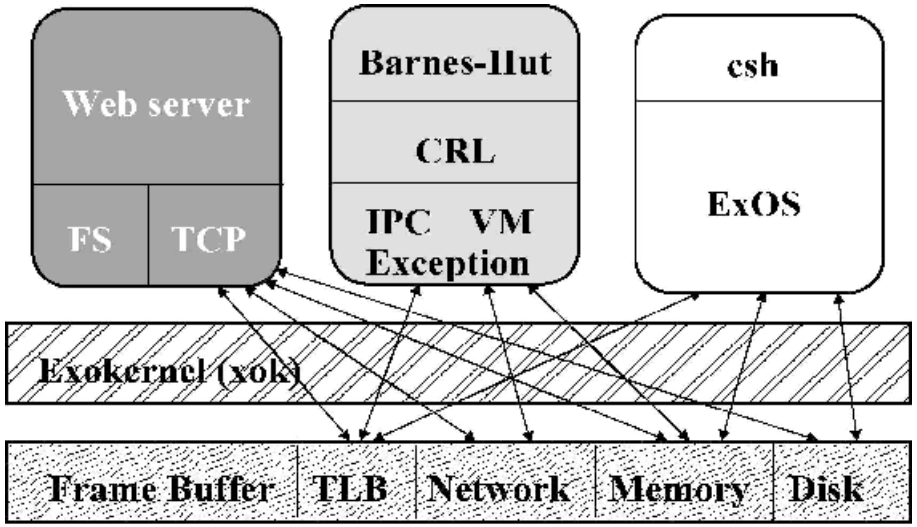
Exokernel

- Microkernel and monolithic emphasize on providing an abstraction of hardware to apps
  - Example: read/write services for files (disk)
- Exokernel
  - No emphasis on abstraction at kernel level
  - Developers of specific applications know better than kernel developers how to use the hardware
  - Exokernel takes OS out of kernel and puts it into application (libraries)
  - Separates protection from management



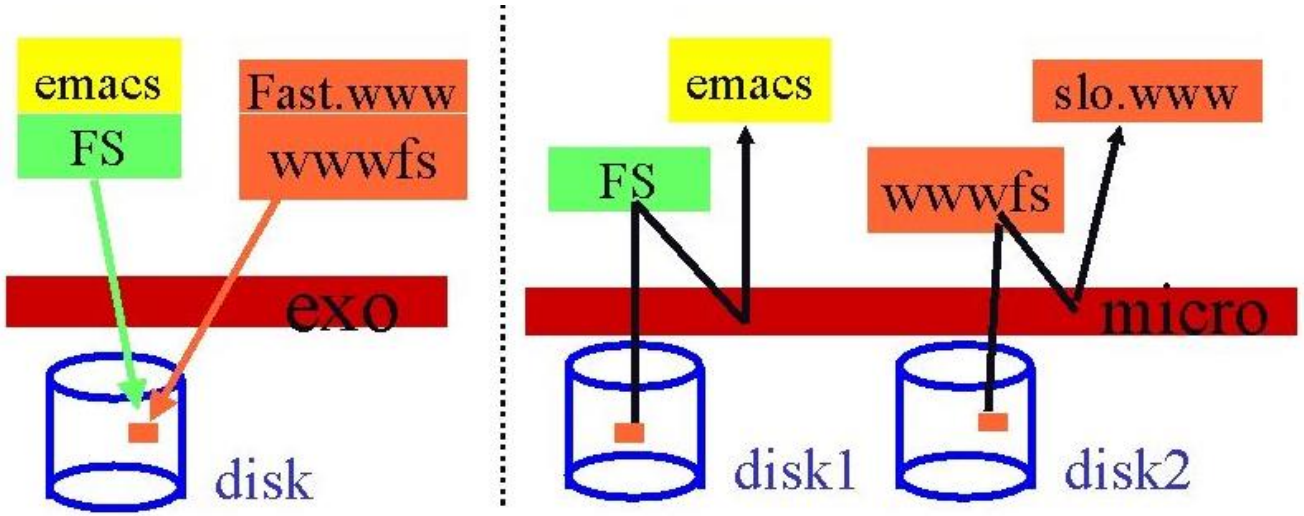
Exokernel

- Exokrenel
  - A thin kernel that just perform multiplexing of hardware resources
    - *Memory, TLB, Disk, Network, Frame Buffer, etc*
  - More complex management is inside application lib
    - *FS, TCP, IPC, etc*



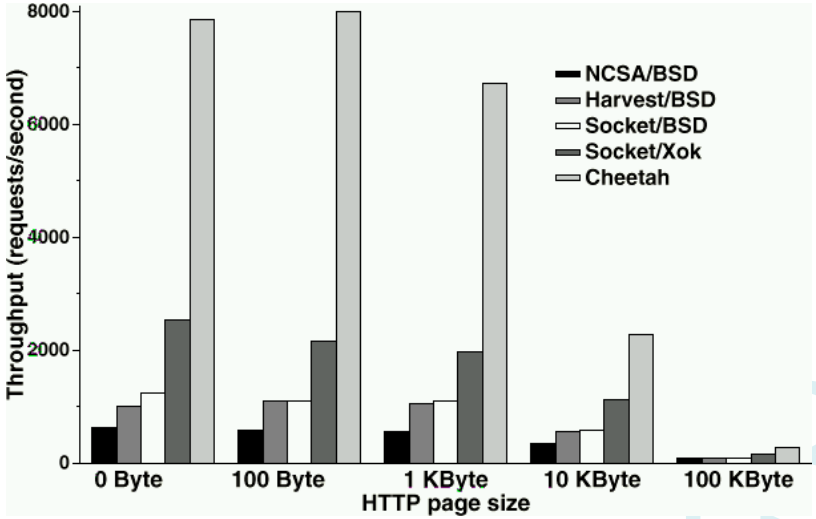
Exokernel

- Exokernel systems can be tailored for specific applications
  - High performance
  - Small software footprint
- Example: Cheetah webserver



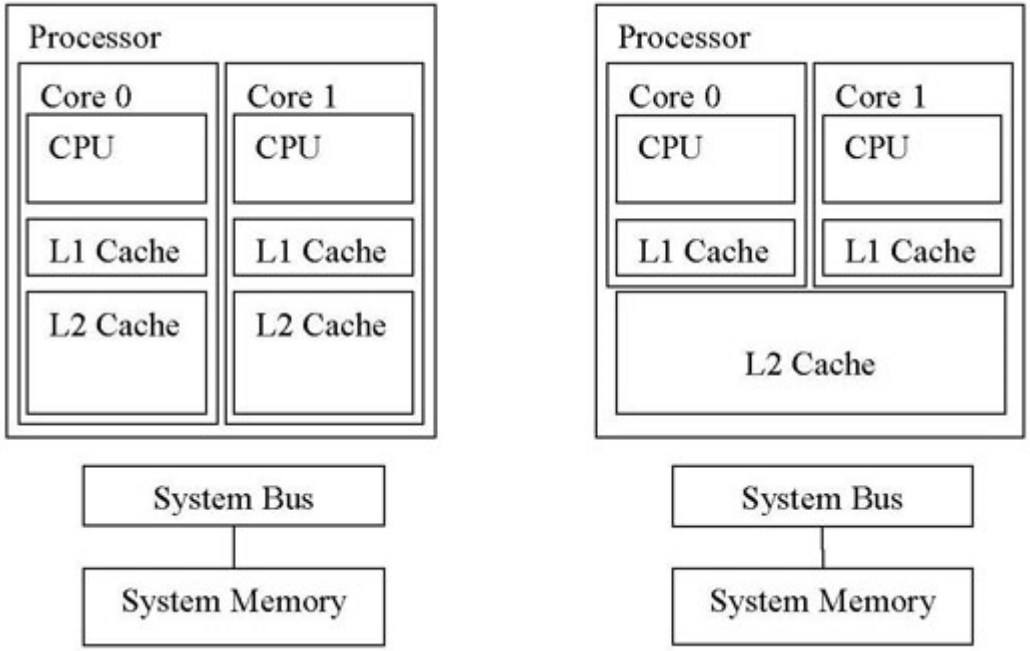
Exokernel Example

- Exokernel example
  - Cheetah web server
- Traditional webserver: 6 buffer operations
  - read() - copy from disk to kernel buffer
  - read() - copy from kernel to user buffer
  - checksum
  - send() - user buffer to kernel buffer
  - send() - kernel buffer to device memory
- Exokernl: 2 operations
  - Copy from disk to memory (packet bodies not files are stored on disk!)
  - checksum
  - Copy from memory to network device

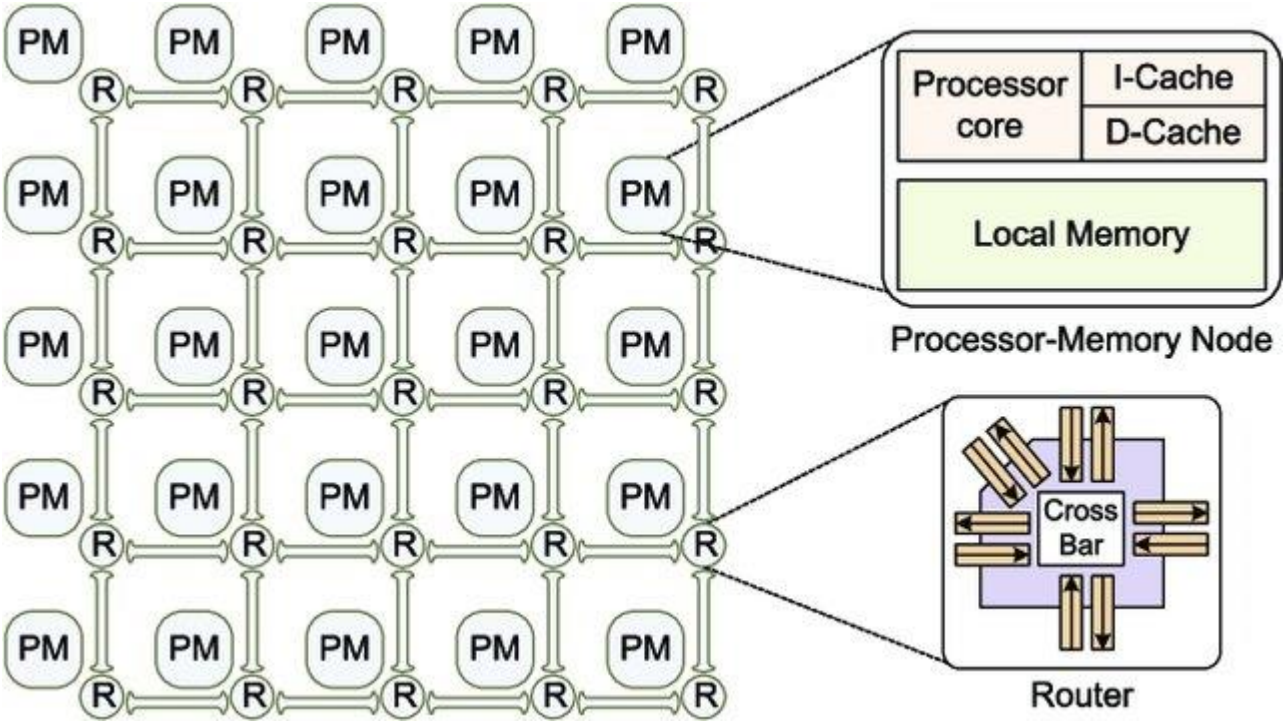


Multikernel

Multicore CPU

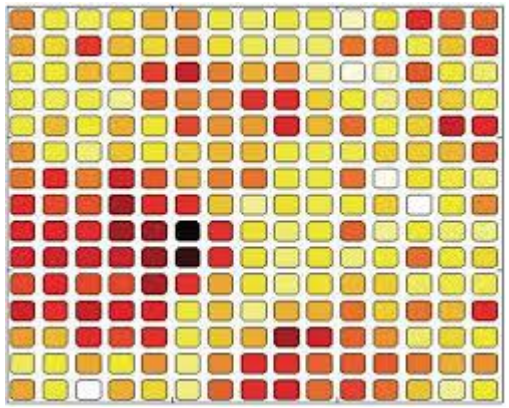


Manycore CPU



Multicore/Manycore OS Challenges

- It is difficult to share a data structure among many cores
  - Locks/Atomic instructions => Scalability problem
  - Requires solution at application level
    - *For example, partition data, use lock-free structures, etc*
    - *Requires tremendous engineering*
- Why?
  - Implicit sharing (via shared memory) is troublesome in manycores
  - It is expensive to move cache lines
  - Congestion of interconnects



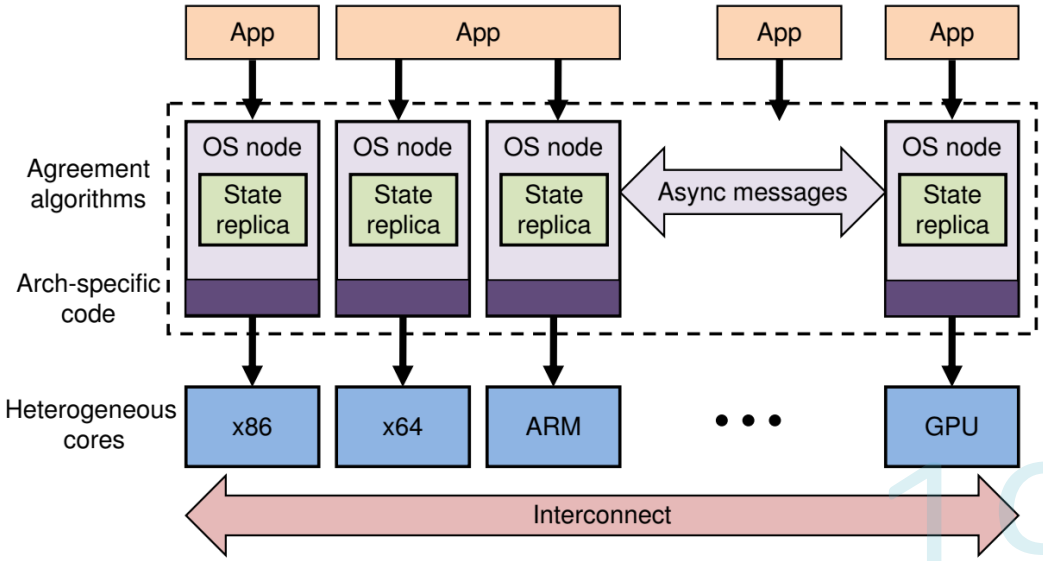
Adopted from OS course by Junfeng Yang https://www.cs.columbia.edu/~junfeng/12sp-w4118

Solution: Multikernel

- Multikernel
  - Kernel for multicores
- Multikernel idea
  - Use explicit sharing via messaging (instead of shared memory)
  - No shared data structure
  - Want to see other core's data? Send a message to it!
  - Advantages
    - *Scalable*
    - *Portable (For example for heterogeneous cores)*
    - *Good match for future if new manycores drop support for shared memory coherency*

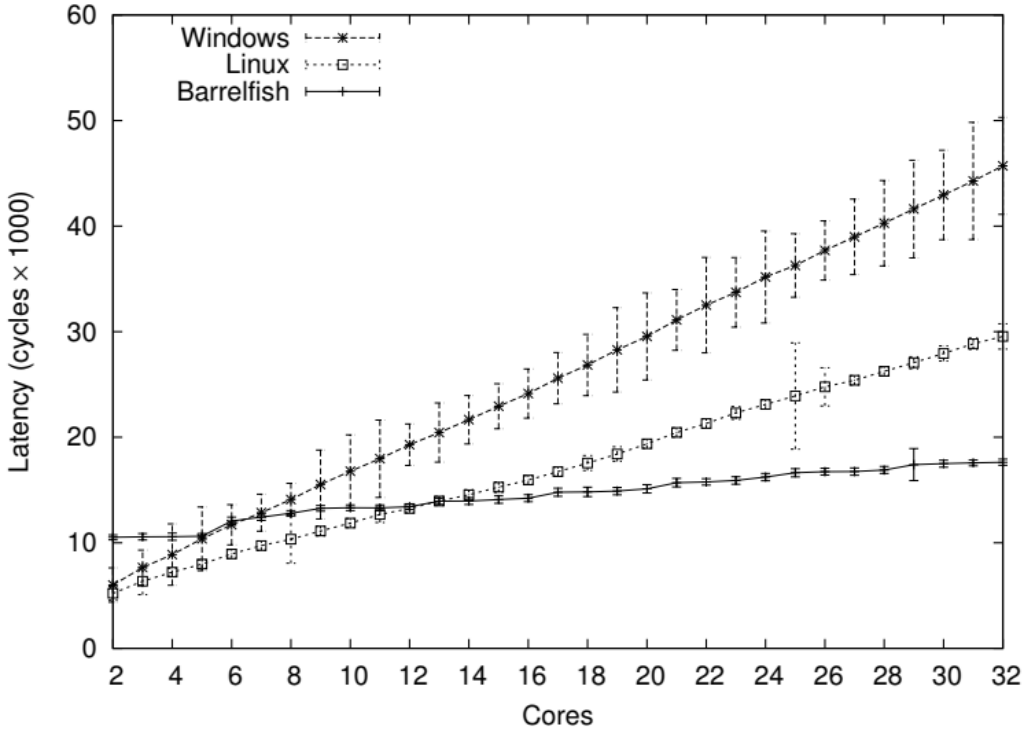
Challenge: global state

- OS must manage global state among cores
  - Example: Page table of a process
  - If a TLB is modified in a core, all cores should update their TLB
    - *AKA TLB Shootdown*
- Solution (in absence of shared data structure)?
  - Replicate the global state
  - Read => OK, low latency
  - Update =>
    - *Update local copy => OK, low latency*
    - *Distribute the update to other cores => !*



Case study: TLB Shootdown

- Need to keep TLB state of a process consistent among cores
- Conventional method (Linux/Windows)
  - Use shared memory for communications
- Multikernel approach
  - Use message passing for communications



References

- Liedtke, Jochen. *On micro-kernel construction*. Vol. 29. No. 5. ACM, 1995.
- Engler, Dawson R., and M. Frans Kaashoek. *Exokernel: An operating system architecture for application-level resource management*. Vol. 29. No. 5. ACM, 1995.
- Madhavapeddy, Anil, et al. "Unikernels: Library operating systems for the cloud." *Acm Sigplan Notices*. Vol. 48. No. 4. ACM, 2013.
- Baumann, Andrew, et al. "The multikernel: a new OS architecture for scalable multicore systems." *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*. ACM, 2009.