PES UNIVERSITY ONLINE

OPERATING SYSTEMS

Computer-System Architecture, OS Structure and Operations

Suresh Jamadagni
Department of Computer Science

Slides Credits for all the PPTs of this course

- The slides/diagrams in this course are an **adaptation**, **combination**, and **enhancement** of material from the following resources and persons:

1. Slides of Operating System Concepts, Abraham Silberschatz, Peter Baer Galvin, Greg Gagne - 9th edition 2013 and some slides from 10th edition 2018
2. Some conceptual text and diagram from Operating Systems - Internals and Design Principles, William Stallings, 9th edition 2018
3. Some presentation transcripts from A. Frank – P. Weisberg
4. Some conceptual text from Operating Systems: Three Easy Pieces, Remzi Arpaci-Dusseau, Andrea Arpaci Dusseau

OPERATING SYSTEMS

Computer-System Architecture, OS Structure & Operations

Suresh Jamadagni
Department of Computer Science

Computer-System Architecture

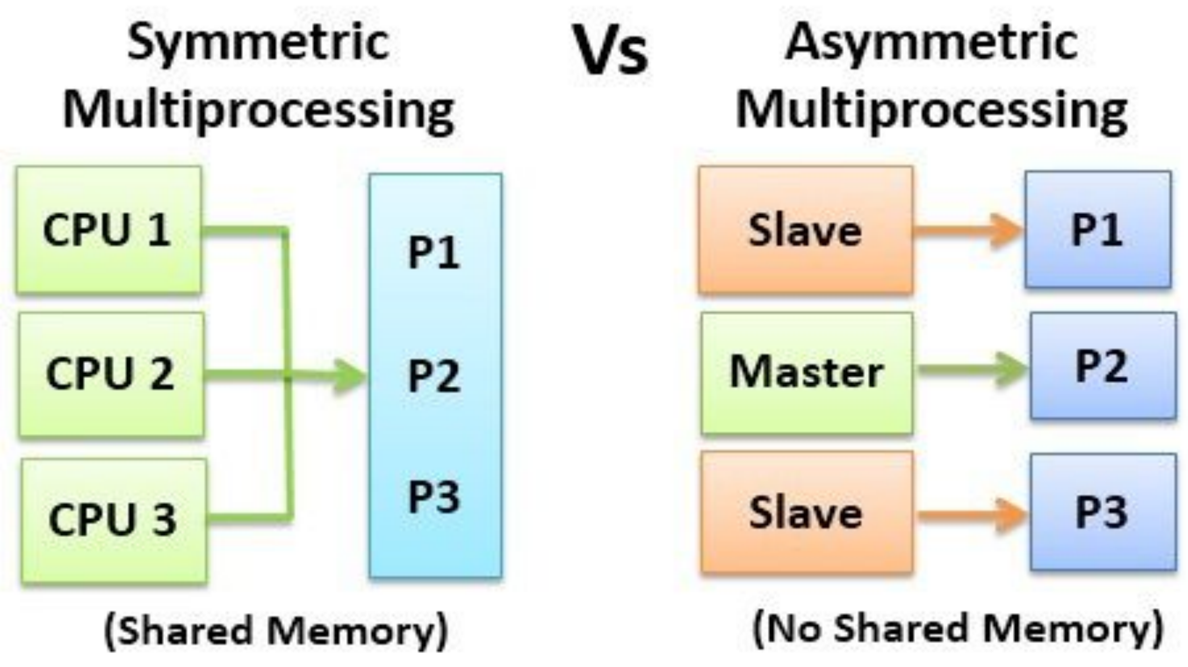
- Most systems use a single general-purpose processor
  - Most systems have other special-purpose processors as well.
  - Device specific processors like disk, keyboard, graphic controller
  - Special-purpose processors run a limited number of instructions
  - Special-purpose processors are low-level components built into the hardware
  - Managed by OS.
  - OS monitors the status.
- Example Disk controller microprocessor
  - Receives sequence of requests from CPU.
  - Implements its own disk queue and scheduling algorithm
  - Relieves the main CPU of the overhead of disk scheduling.

Computer-System Architecture

- **Multiprocessors** systems growing in use and importance
  - Also known as **parallel systems**, **tightly-coupled systems**
  - Advantages include:
    - **Increased throughput**
    - **Economy of scale**
    - **Increased reliability** – graceful degradation or fault tolerance

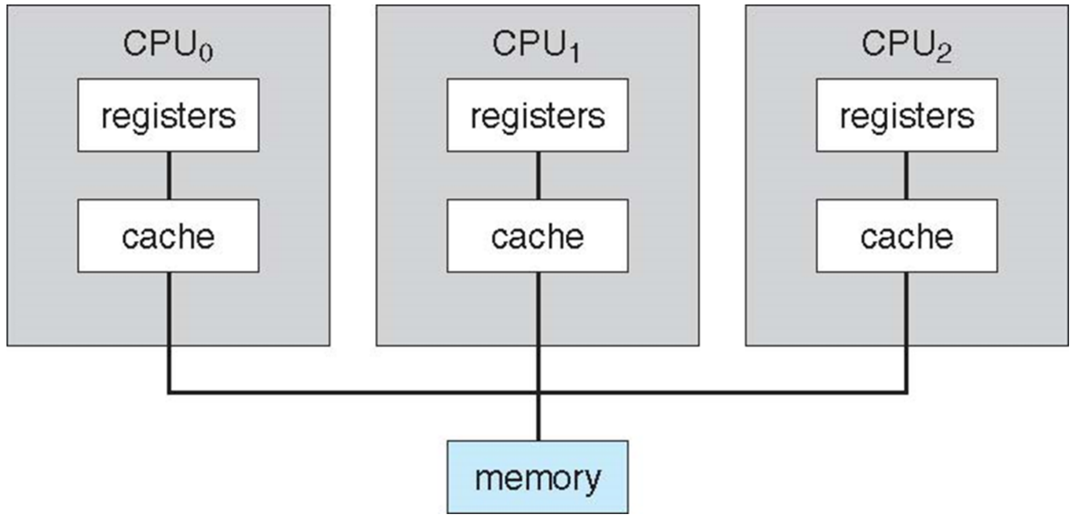
Two types of Multiprocessor Systems

1. **Asymmetric Multiprocessing** – each processor is assigned a specific task.
2. **Symmetric Multiprocessing** – each processor performs all tasks



Symmetric Multiprocessing Architecture

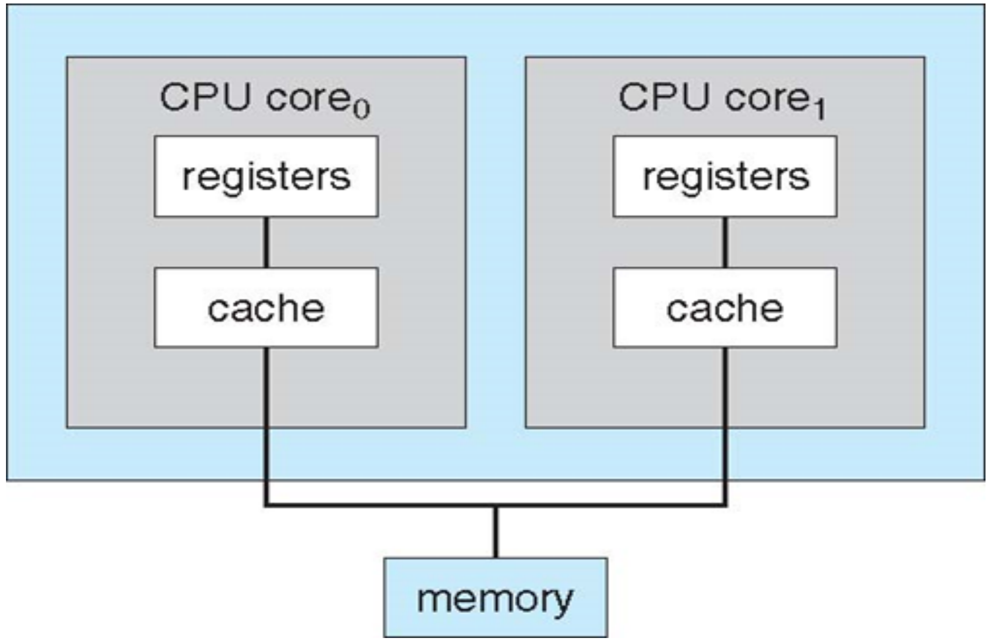
- In SMP all processors are peers; no boss–worker relationship exists between processors.
- Each processor has its own set of registers, as well as a private or local cache.
- All processors share physical memory.



OPERATING SYSTEMS

Multi-Core Design

- A recent trend in CPU design is to include multiple computing cores on a single chip. Such multiprocessor systems are termed **multicore**.
- More efficient than multiple chips with single cores because on-chip communication is faster than between-chip communication.
- One chip with multiple cores uses significantly less power than multiple single-core chips.



A dual-core design with two cores placed on the same chip

Command to know the number of cores, cache details

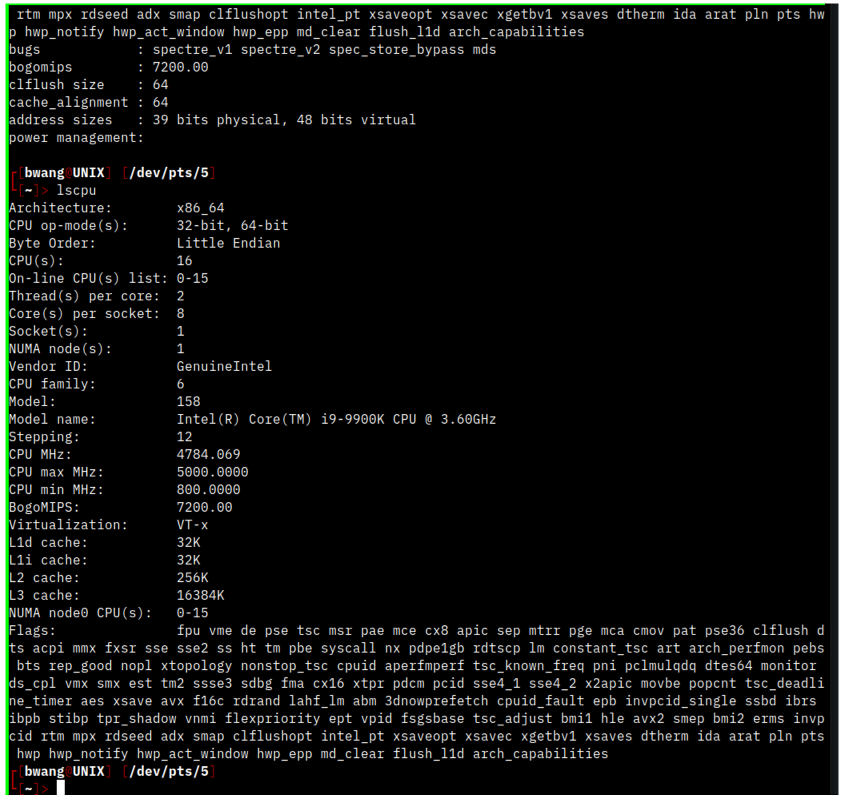
```
$cat /proc/cpuinfo|more
```

OPERATING SYSTEMS

Multi-Core Design

Command to know the number of cores, cache details

$more /proc/cpuinfo



```
 rtm mpx rdseed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln pts hw
p hwp_notify hwp_act_window hwp_epp md_clear flush_l1d arch_capabilities
bugs            : spectre_v1 spectre_v2 spec_store_bypass mds
bogomips        : 7200.00
clflush size    : 64
cache_alignment : 64
address sizes   : 39 bits physical, 48 bits virtual
power management:

┌[bwang@UNIX] [/dev/pts/5]
└[~]> lscpu
Architecture:        x86_64
CPU op-mode(s):      32-bit, 64-bit
Byte Order:          Little Endian
CPU(s):              16
On-line CPU(s) list: 0-15
Thread(s) per core:  2
Core(s) per socket:  8
Socket(s):           1
NUMA node(s):        1
Vendor ID:           GenuineIntel
CPU family:          6
Model:               158
Model name:          Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz
Stepping:            12
CPU MHz:             4784.069
CPU max MHz:         5000.0000
CPU min MHz:         800.0000
BogoMIPS:            7200.00
Virtualization:      VT-x
L1d cache:           32K
L1i cache:           32K
L2 cache:            256K
L3 cache:            16384K
NUMA node0 CPU(s):   0-15
Flags:               fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush d
ts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs
 bts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf tsc_known_freq pni pclmulqdq dtes64 monitor
ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadli
ne_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb invpcid_single ssbd ibrs
ibpb stibp tpr_shadow vnmi flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invp
cid rtm mpx rdseed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln pts
 hwp hwp_notify hwp_act_window hwp_epp md_clear flush_l1d arch_capabilities
┌[bwang@UNIX] [/dev/pts/5]
└[~]>
```

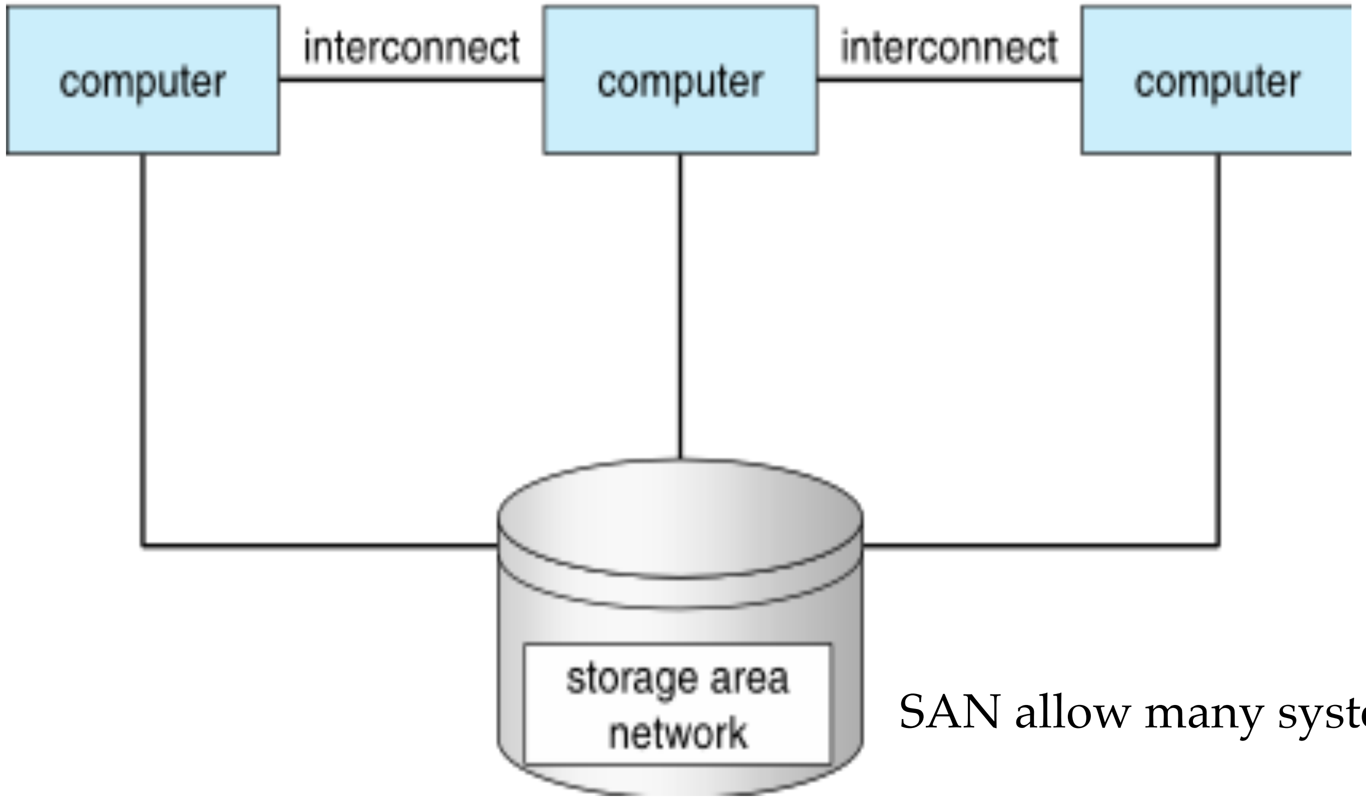
Blade servers

- **Blade servers** are a recent development in which multiple processor boards, I/Oboards, and networking boards are placed in the same chassis.
  - blade-processor board boots independently and runs its own operating system.
  - Some blade-server boards are multiprocessor as well, which blurs the lines between types of computers.
  - In essence, these servers consist of multiple independent multiprocessor systems.

Clustered Systems

- Like multiprocessor systems, but multiple systems working together
  - Usually sharing storage via a **storage-area network (SAN)**
  - Provides a **high-availability** service which survives failures
    - **Asymmetric clustering** has one machine in hot-standby mode
    - **Symmetric clustering** has multiple nodes running applications, monitoring each other
  - Some clusters are for **high-performance computing (HPC)**
    - Applications must be written to use **parallelization**
  - Some have **distributed lock manager** (**DLM**) to avoid conflicting operations (Ex: when multiple hosts access the same data on shared storage)

Clustered Systems

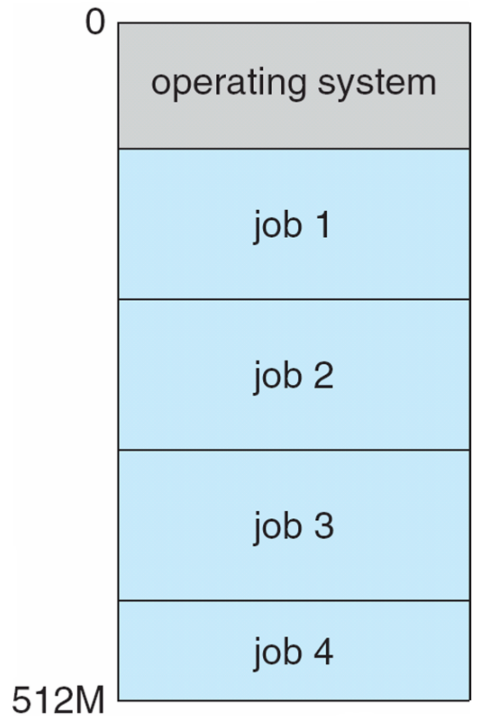


SAN allow many systems to attach to a pool of storage

General structure of a clustered system.

Operating-System Structure - Multiprogramming

- **Multiprogramming** (**Batch system**) needed for efficiency
  - Single user cannot keep CPU and I/O devices busy at all times
  - Multiprogramming organizes jobs (code and data) so CPU always has one to execute
  - A subset of total jobs in system is kept in memory
  - One job selected and run via **job scheduling**
  - When it has to wait (for I/O for example), OS switches to another job



Operating-System Structure - Multitasking

- **Timesharing** (**multitasking**) is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating **interactive** computing
  - **Response time** should be < 1 second
  - Each user has at least one program executing in memory ⇨ **process**
  - If several jobs ready to run at the same time ⇨ **CPU scheduling**
  - If processes don't fit in memory, **swapping** moves them in and out to run
  - **Virtual memory** allows execution of processes not completely in memory

Operating-System Operations

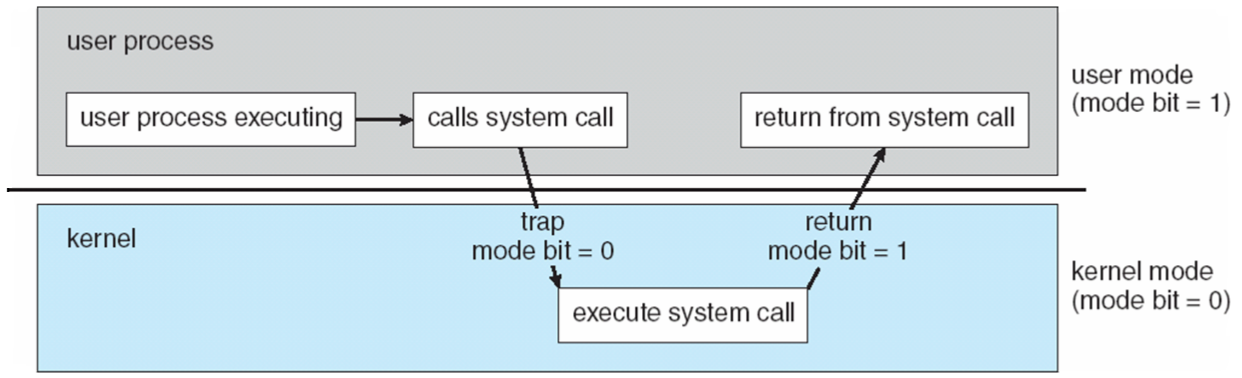
- **Interrupt driven** (hardware and software)
  - Hardware interrupt by one of the devices
  - Software interrupt (**exception** or **trap):**
    - Software error (e.g., division by zero)
    - Request for operating system service
    - Other process problems include infinite loop, processes modifying each other or the operating system

Dual-Mode and Multimode Operation

- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as **privileged**, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user
- Increasingly CPUs support multi-mode operations
  - i.e. **virtual machine manager** (**VMM**) mode for guest **VMs**

Transition from user to kernel mode

- When a trap or interrupt occurs, hardware switches from user mode to kernel mode (changes the state of the mode bit to 0).
- When the request is fulfilled, the system always switches to user mode (by setting the mode bit to 1) before passing control to a user program.



Timer

- Timer to prevent infinite loop / process hogging resources
  - Timer is set to interrupt the computer after a specified period (fixed 1/60 sec or variable 1 msec to 1 sec)
  - A **variable timer** is generally implemented by a fixed-rate clock and a counter.
  - Operating system sets the counter (privileged instruction)
  - Every time the clock ticks, the counter is decremented.
  - When counter reaches zero, an interrupt occurs
  - Timer can be used to prevent a user program from running too long (terminate the program)

PES UNIVERSITY ONLINE

THANK YOU

Suresh Jamadagni
Department of Computer Science Engineering
sureshjamadagni@pes.edu