Goal of this section: key concepts you should understand

- Not just a summary of lectures
- Slides coverage and final exam topics are not bijective

Goal of CSE 451: tools for life

Goal of your life: ????
Thread management

Queues
- Why do thread libraries make use of queues?

Synchronization
- What are the mechanisms for protecting critical sections, how do they work, and when should one be used over another?

Preemption
- What is preemption and how does the process of one thread preempting another work?
Memory management

**Purposes:**
- Resource partitioning / sharing
- **Isolation**
- Usability

**Paging**

**Segmentation**
Virtual memory

* What happens on a virtual memory access?
Virtual memory

What happens on a virtual memory access?
- Address translation: who performs it?
  - Page table lookup
  - Translation Lookaside Buffer (TLB)
- Page fault?
  - Page replacement
  - Process/queue management

How does all of this overhead pay off?
- Locality! Both temporal (in time) and spatial (nearby).
Virtual memory

Note: Each process has its own page table!
Page replacement

- Algorithms:
  - Belady, FIFO, LRU, LRU clock / NRU, random, working set...
  - Local vs. global

- How/why are any of these better or worse than the others?

- What happens when paging goes wrong?
  - Thrashing, 10-year old computers running XP?
Advanced virtual memory

★ What problem does a TLB address?

★ What problem do two-level page tables address?
  ★ What’s the key concept?
**Advanced virtual memory**

- What problem does a TLB address?
  - Increases speed of virtual address translation

- What problem do two-level page tables address?
  - What’s the key concept?
    - Indirection
Secondary storage

* Memory forms a *hierarchy*

* Different levels of disk abstraction:
  * Sectors
  * Blocks
  * Files

* What factor most influences the ways that we interact with disks?
Secondary storage

- Memory forms a hierarchy
- Different levels of disk abstraction:
  - Sectors
  - Blocks
  - Files
- What factor most influences the ways that we interact with disks?
  - Latency
Memory hierarchy

- CPU registers: 128 bytes, 1 cycle
- L1 cache: 32 KiB, 4 cycles
- L2 cache: 4 x 256 KiB, 11 cycles
- L3 cache: 8 MiB, 39 cycles
- Primary Memory: 8 GiB, 100 cycles
- Secondary Storage: 1 TiB, 30 million cycles
- Tertiary Storage: 1 PiB, ??? cycles

- Each level acts as a cache of lower levels
- (Stats more or less for Core i7 3770)
What does a file system give you?

- Useful abstraction for secondary storage
- Organization of data
  - Hierarchy of directories and files
- Sharing of data
File system internals

- Directories
- Directory entries
- Inodes

Files:
- One inode per file
- Multiple directory entries (links) per file
Inode-based file system

- Sequence of steps when I run `echo "some text" > /home/jay/file.txt`?
  - Open file:
    - Get inode for / -> get data block for /
    - Read directory entry for / -> get inode for /homes
    - Repeat... -> get data block for file.txt, check permissions
  - Write to file:
    - Modify data block(s) for file.txt in buffer cache
  - Close file:
    - Mark buffer as dirty, release to buffer cache
    - Kernel flushes dirty blocks back to disk at a later time
Other file systems

What problem does each of these address?

- BSD Unix fast file system (FFS):
  - Performance: smarter physical disk layout

- Journaling file systems (JFS):
  - Reliability: transactions prevent inconsistencies after crash

- Berkeley log-structured file system (LFS):
  - Performance: even smarter physical disk layout?
RAID

- **Striping**: read/write from multiple disks simultaneously
  - Improves performance
  - Hurts reliability

- **Parity**: store redundant information to allow data recovery after disk failures
  - Improves reliability
  - Hurts performance

12/5/2013
How should the OS provide access to physical hardware to user processes?

- Multiplexing
- Mutual exclusion

UNIX / Linux device driver model

Virtual devices, and what they can do for you

- FUSE
Networking

- Layering
- Encapsulation
RPC

**Benefits:**
- Low-level details taken care of for you
- Natural interface

**Implementation issues:**
- Network failures / retries
- Architecture differences
- Performance
Distributed file systems

Why do we want them?
- Location independence
- Large-scale data sharing

Why are they hard?
- Consistency
- Replication
- Performance

Understand the target workloads
Distributed systems

**Scalability**
- Limited by sharing
  - How does this relate to multi-core CPUs?
- Do more nodes equal more performance?
- How do companies like Amazon, Facebook, Google, Microsoft, etc. parallelize workloads?
Virtual machine monitors

- VMM is an additional layer between OS and hardware
- Can interpose on instruction execution, memory accesses, I/O requests, and network communication