Outline of lectures to date

With references to relevant sections in the book. Note that not all topics will be covered in the exam.

Lecture 1 - What is an operating system? Overview of a simple hardware and program model, introduction of the idea of a system call abstraction between operating system and program. (Silbershatz et al. 1.3, 1.4, 2.1, 2.3)

Lecture 2 - MSDOS vs. UNIX and Linux, context switching. The concept of computation state being stored in stack contents + registers + program counter. (3.1, 3.3, 4.1)

Lecture 3 - Concurrency models - threads, processes, virtual machines. Lighter-than-threads models - events, pseudo-threads. Kernel vs. user threads. (4.2)

Lecture 4 - Review of subroutine linkage, mutual exclusion, Peterson's algorithm, spinlock. (http://en.wikipedia.org/wiki/Call_stack, 6.2, 6.3, 6.4)

Lecture 5 - More about mutexes, the producer/consumer problem, semaphores, rendezvous, monitors. Mesa semantics for monitors. (6.5, 6.6, 6.7)

Lecture 6 - Pointer and array equivalence in C. Equivalence of mutexes and semaphores by construction, equivalence of monitors and semaphores. Java monitors, POSIX threads, transactional memory. Consistency in the face of concurrency vs. failure: file systems and databases are concerned w/ failure consistency, use locking, ordering, and journaling. (equivalence - see notes. rest - 6.9)

Lecture 7 - Typical process memory map, object file format - code, data, BSS, stack. Static/dynamic linking and loading. (mostly in notes. Also 8.1)

Lecture 8 - Deadlock, the 4 conditions for deadlock. They're not strictly necessary, despite what the book says. FSM view of multiple processes interacting, deadlock as a dead state in the FSM. Lock ranking to ensure freedom from deadlock. (chapter 7 minus resource graphs and banker's algorithm)

Lecture 9 - Queueing theory, Little's formula, M/M/1 queue, rho (load factor) = lambda (arrival rate) / mu (service rate), mean inter-arrival time / service time = 1/lambda, 1/mu. E(#waiting jobs) = rho/(1-rho) for M/M/1 queue. (class notes)

Lecture 10 - Markov models, specifically discrete Markov models with memoryless transitions; balance equations for Markov model and solving for state occupancy. Simulation, based on synthetic input or measured traces. (class notes)

Lecture 11 - Fangfei's preemptive kernel question. Virtual memory - base and bounds, paged. 32 bit 4K page 2-level page table. TLB. (8.3, 8.4, 8.5.1)

Lecture 12 - Page attributes - read, write, sometimes execute, user/supervisor. Shared pages. All addresses are virtual when virtual addressing is on. Virtual memory gives: language-agnostic allocator, no external fragmentation, lazy allocation, demand paging. It's useful even if you never page to/from disk.
Lecture 13 - Memory allocators - interval list (first/best/worst fit), binary buddy, general extent allocators. Slab allocator*. (8.3, 9.8.1, 9.8.2)


Lecture 15 - Page replacement. NUMA tangent*. Stack algorithms for replacement - "inclusive". (i.e. memory always contains contents of same algorithm & trace with smaller memory) FIFO + 2nd chance = CLOCK. Note that amortized clock overhead is small, as clock speed adjusts to memory pressure and dirtying rate. Working set vs. global replacement. Cyclic patterns, failure of LRU on. Garbage collection, open questions re: GC and page replacement. Miss ratio curve (MRC). (9.4, 9.6.2)

Lecture 16 - Virtualization. Impossibility of running conventional OS over conventional OS. Total software emulation of CPU. Virtual I/O by page fault & emulate. Popek & Goldberg, "sensitive" vs "innocuous" instructions. Trap and emulate for virtualization of sensitive instructions. Virtualizing x86 - software emulate all kernel-mode, emulate + JIT ("binary translation") for kernel mode, hardware support via 3rd privilege level (Intel VT, AMD SVM). (2.8, but mostly class notes)

Outline by topic

- The basic machine and process model, including stack, subroutine linkage, memory map, memory-mapped I/O.
- Process to operating system abstraction layer - i.e. system calls
- Threads and context switching.
- Mutual exclusion - mutexes, semaphores, and monitors.
- Deadlock - criteria, lock ranking, thread interaction as FSM, deadlock as dead state.
- Queuing - Little's formula, M/M/1 queue, E[N] for M/M/1, Markov model.
- Page tables - canonical 32-bit page table (10/10/12), TLB. Shared pages.
- Page fault handling - fix page tables & retry instruction, used to implement: demand loading, lazy allocation, copy-on-write.
- Allocators - fragmentation (internal & external), buddy/extent algorithms.
- Page replacement - random, FIFO, LRU, OPT, clock.
- Virtualization - Popek & Goldberg criteria, trap & emulate implementation for virtualizable CPUs, full emulation / binary translation / 3rd privilege level for non-virtualizable CPUs (i.e. Intel x86).

Midterm Exam questions

Finally, of this material the mid-term is going to consist of 4 questions, of which you will choose 3. The questions will be on the following topics:

- Threads, processes, and context switching.
- Synchronization and deadlocks.
- Queueing and/or Markov models.
- Virtual memory.