

Steve Martin  
CS 162 Spring 2004  
Midterm I Review Topics

1. What is an operating system?
  - a. Batch systems
  - b. Uniprogramming vs. Multiprogramming
2. Function of an operating system
  - a. Coordinator
    - i. Concurrency
    - ii. I/O
    - iii. Memory
    - iv. Files
    - v. Networks
  - b. Extended Machine
    - i. Present nice interface to messy hardware
3. Operating system structure
  - a. What are the components of an OS?
  - b. What kinds of services does the OS provide?
  - c. What are the design goals for the OS?
  - d. System structure
    - i. "Simple" structure
    - ii. Layered approach
    - iii. Microkernel
4. Processes
  - a. What are they?
  - b. How are they made?
  - c. How are they represented in an OS?
    - i. E.g. what state do they have?
  - d. How many processes can run on a CPU at once?
  - e. What are process states?
5. Threads
  - a. What are they?
  - b. How are they made?
    - i. How are they joined?
  - c. What state do they maintain?
  - d. How many can run on a CPU at once?
  - e. How do they differ from processes?
  - f. Why separate threads from processes?
  - g. What is different between a thread and a process context switch?
  - h. What does the state diagram for a thread look like?
  - i. How does a thread go from inactive to running?
    - i. What needs to happen?
  - j. Can we use both threads and processes?
    - i. Why would we do this?
6. Cooperating threads

- a. What is the difference between multiprocessing and multiprogramming?
  - b. What is the definition of ‘run concurrently’?
  - c. What are independent threads?
  - d. What are cooperating threads?
7. Exceptions
- a. What is an interrupt?
    - i. What happens when one occurs?
  - b. What is a trap/exception?
    - i. What happens when one occurs?
8. Atomicity
- a. What is an atomic operation?
  - b. Why are they important?
  - c. How can you ensure an instruction is atomic?
    - i. How about with multiple CPUs?
9. Synchronization
- a. What is synchronization?
  - b. What is a critical section?
  - c. What is mutual exclusion?
  - d. What is busy-waiting? (or ‘spinning’)
  - e. What is a lock?
    - i. What are its properties?
  - f. What is a semaphore?
    - i. What are its properties?
  - g. What is a condition variable?
    - i. What are its properties?
    - ii. How are these different from semaphores?
  - h. What is a monitor?
    - i. What is a Mesa-style monitor?
    - ii. What is a Hoare-style monitor?
  - i. What is a mutex?
  - j. Can you build one kind of synchronization primitive out of another?
  - k. What is important about test-and-set and swap?
  - l. What language support exists for synchronization?
10. Deadlock
- a. What is deadlock?
    - i. What are the four conditions of deadlock?
  - b. What are examples of preemptible and non-preemptible resources?
  - c. What are the two approaches to the deadlock problem?
  - d. What are ways to prevent deadlock?
    - i. What is a ‘safe state’?
  - e. Bankers Algorithm
  - f. Graph Algorithm
    - i. What is a resource allocation graph?
11. Scheduler
- a. What is a dispatcher/scheduler?
    - i. What are its responsibilities?

- b. When does it run?
- c. How does it maintain control of the CPU?
  - i. How does it put a process on the CPU?

## 12. Scheduling

- a. What are scheduling goals?
  - i. What is flow time?
  - ii. What is throughput?
  - iii. What is response time?
  - iv. Why is fairness important?
    - 1. When is it not important?
- b. How are processes scheduled on the CPU?
  - i. FIFO (First in first out)
  - ii. RR (Round robin)
  - iii. SJF (Shortest job first)
  - iv. SRTF (Shortest remaining time first)
  - v. EQ, or MLFQ (Multi-level feedback queues)
- c. How do job characteristics influence scheduling choice?
- d. What is the difference between a preemptive and non-preemptive scheduling policy?
- e. What is an adaptive scheduling policy?

## 13. Protection

- a. Multiprogramming without protection
  - i. How do linkers and loaders work?
  - ii. What are the components of a program in memory?
- b. What are hardware protection modes?
  - i. Why do we need these?
- c. How can a user program access OS services?
  - i. How does the protection mode switch?

## 14. Address translation

- a. Why do we need this?
- b. What is a Virtual Address? What is a Physical Address?
- c. What is internal fragmentation? What is external fragmentation?
- d. What is base and bound relocation?
  - i. What are the advantages and disadvantages?
  - ii. What does base-and-bounds relocation add to context switches?  
Memory lookups?
- e. Segmentation
  - i. What is a segment table?
  - ii. What does segmenting fix that is broken with base and bounds?
  - iii. What is a segment table entry? What does it look like?
    - 1. What is each field for?
  - iv. What is a segment fault?
  - v. What exactly happens when a program tries to read from memory?
  - vi. What does segmentation add to context switches?
  - vii. What is a core map?
- f. Paging

- i. What is a page table?
- ii. What does paging fix that is broken in segmenting?
- iii. What does a page table entry look like?
  - 1. What is each field for?
- iv. What is a page fault?
- v. What exactly happens when a program tries to read from memory?
- vi. What does paging add to context switches?
- vii. Can you page out anything in memory?
- viii. How big is a page table?
- ix. What is 2-level paging?
  - 1. How does this work?
- x. What is an inverted page table?
  - 1. How does this work?
  - 2. Why do you need one?
- g. How do these methods provide protection among processes?
- h. How would you combine segmentation with paging?
  - i. Why would you want to?

#### 15. Caches and TLBs

- a. What is the primary motivation behind using a cache?
- b. What are the different methods for searching a cache?
  - i. What is direct mapped?
  - ii. What is set associative?
  - iii. What is fully associative?
  - iv. What are the advantages and disadvantages of each?
- c. What is a TLB?
  - i. Why do we use them?
- d. What happens to the TLB on a context switch?
- e. What does using a TLB add to handling page faults?
- f. What is the effective access time of a TLB?
  - i. E.g. how is it calculated?