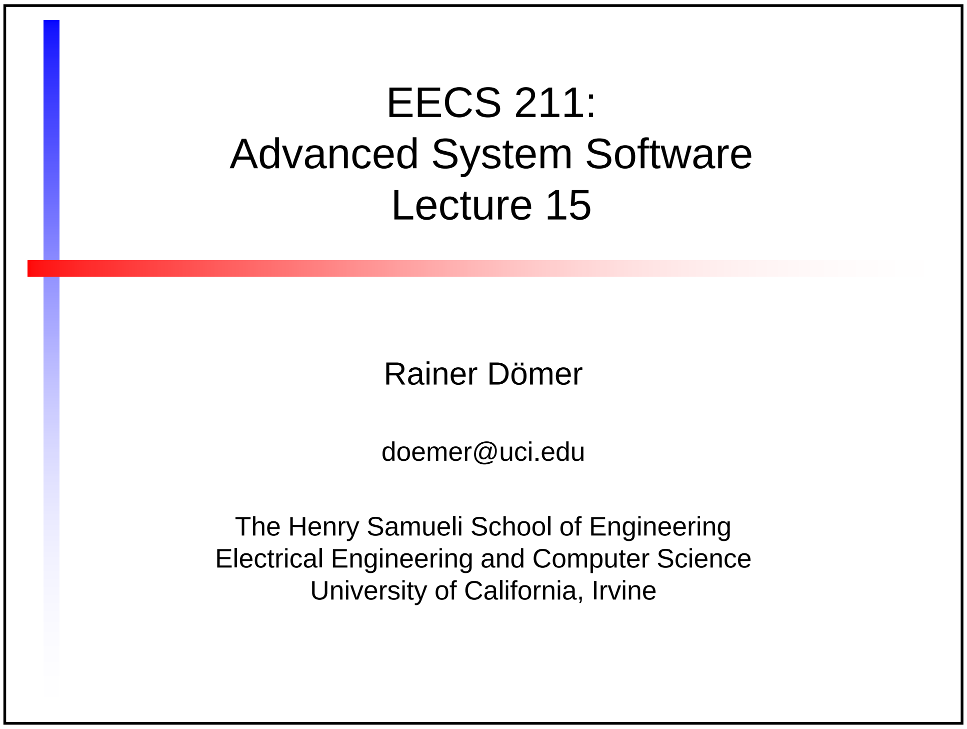
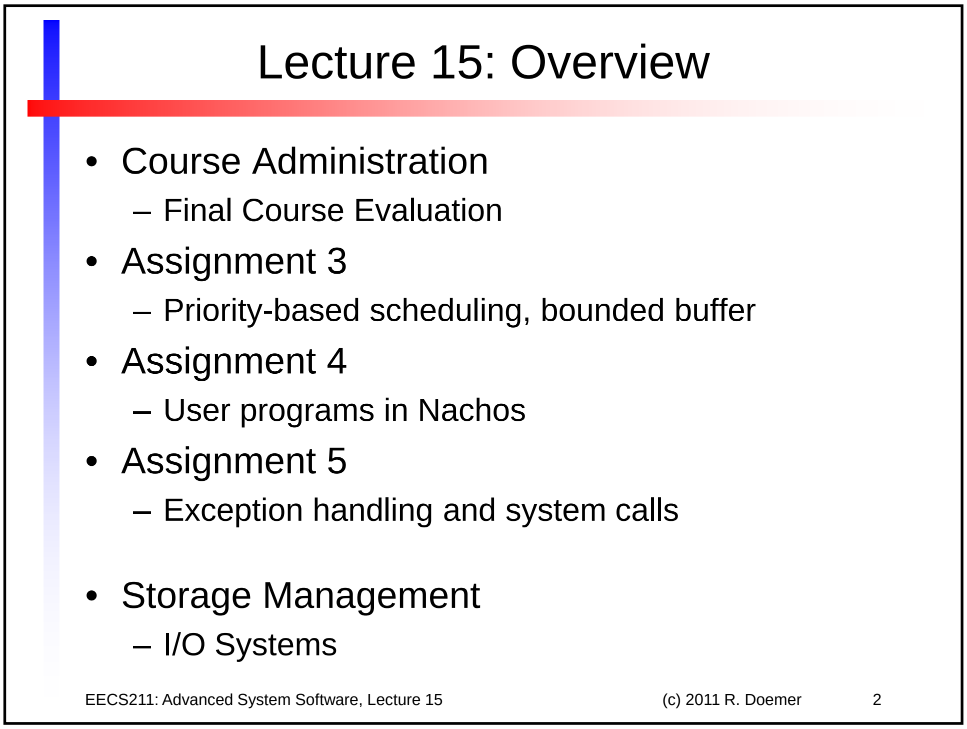


EECS 211:
Advanced System Software
Lecture 15

Rainer Dömer

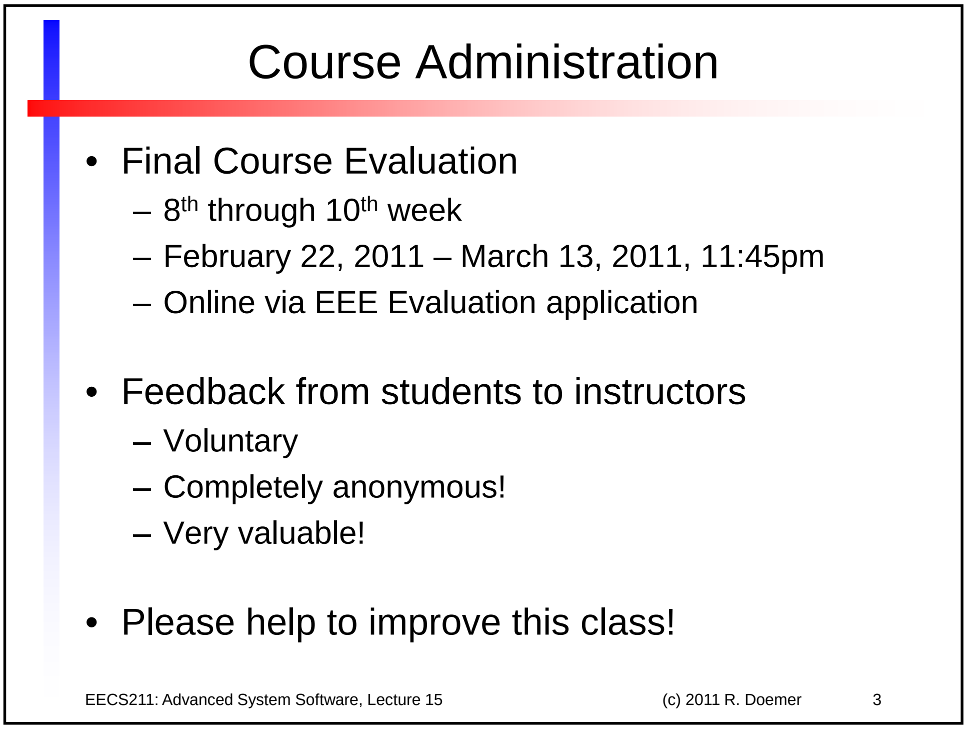
doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine



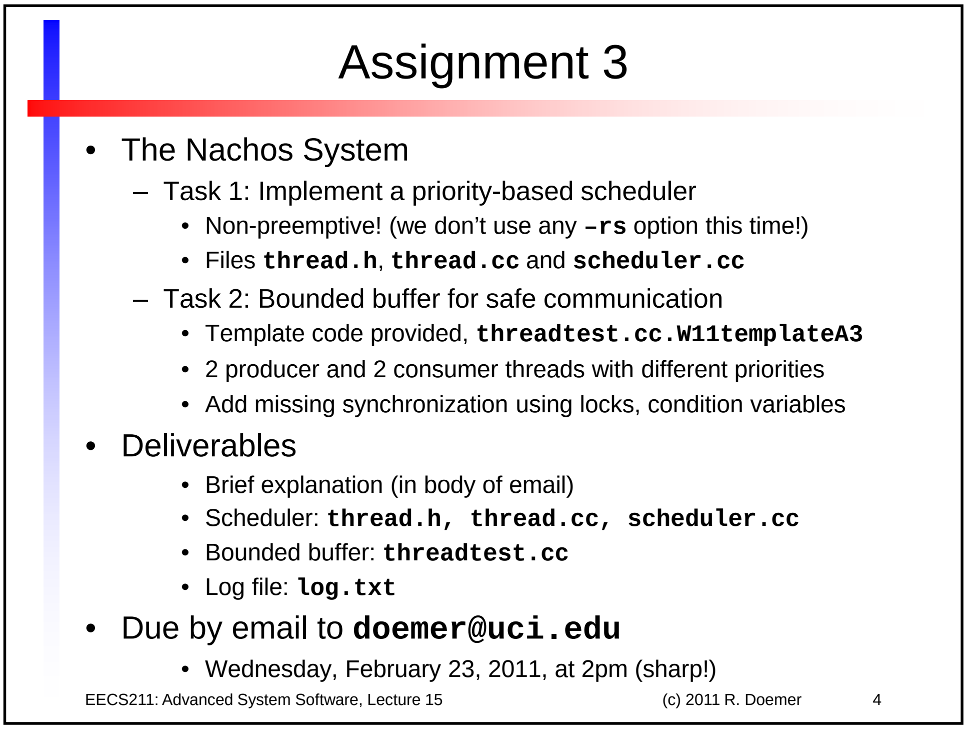
Lecture 15: Overview

- Course Administration
  - Final Course Evaluation
- Assignment 3
  - Priority-based scheduling, bounded buffer
- Assignment 4
  - User programs in Nachos
- Assignment 5
  - Exception handling and system calls
- Storage Management
  - I/O Systems



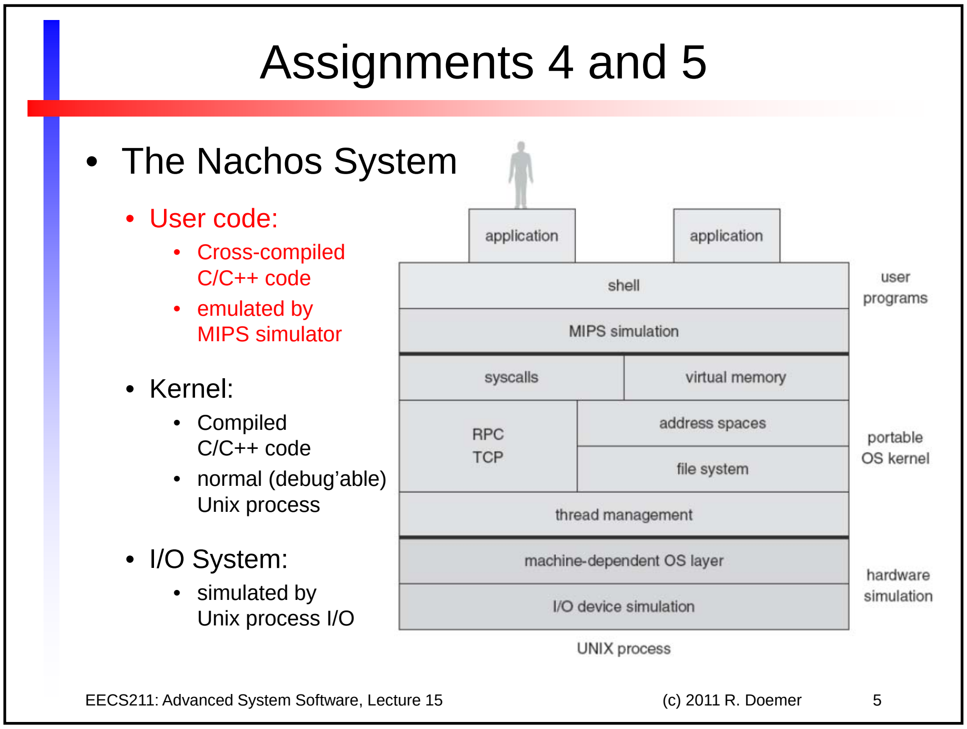
Course Administration

- Final Course Evaluation
  - 8th through 10th week
  - February 22, 2011 – March 13, 2011, 11:45pm
  - Online via EEE Evaluation application
- Feedback from students to instructors
  - Voluntary
  - Completely anonymous!
  - Very valuable!
- Please help to improve this class!



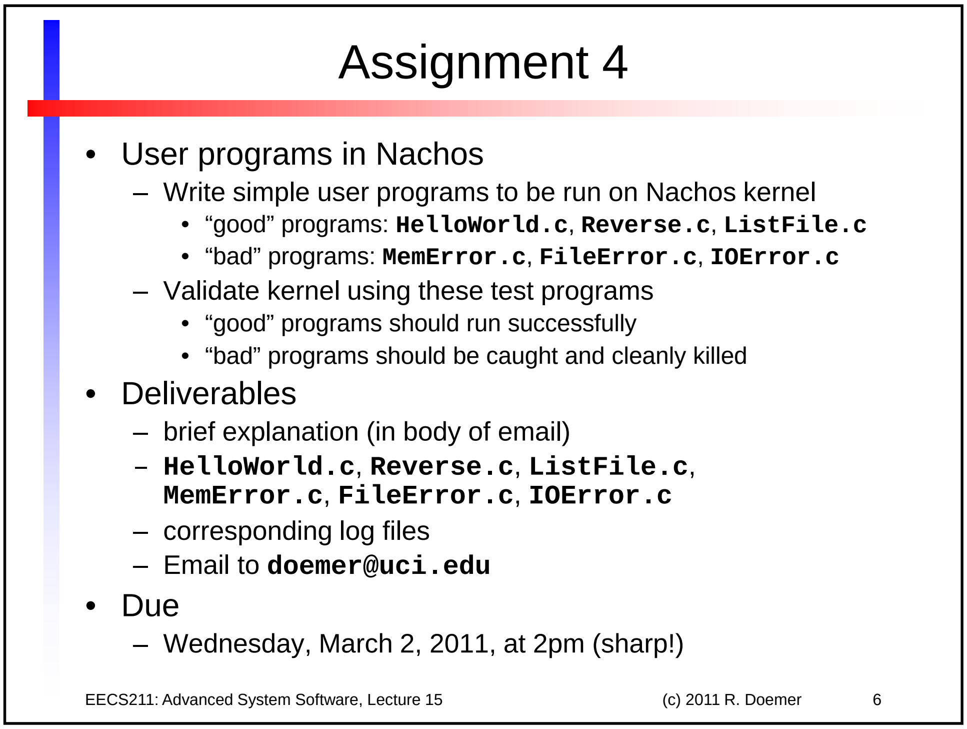
Assignment 3

- The Nachos System
  - Task 1: Implement a priority-based scheduler
    - Non-preemptive! (we don’t use any **`-rs`** option this time!)
    - Files **`thread.h`**, **`thread.cc`** and **`scheduler.cc`**
  - Task 2: Bounded buffer for safe communication
    - Template code provided, **`threadtest.cc.W11templateA3`**
    - 2 producer and 2 consumer threads with different priorities
    - Add missing synchronization using locks, condition variables
- Deliverables
    - Brief explanation (in body of email)
    - Scheduler: **`thread.h, thread.cc, scheduler.cc`**
    - Bounded buffer: **`threadtest.cc`**
    - Log file: **`log.txt`**
- Due by email to **`doemer@uci.edu`**
    - Wednesday, February 23, 2011, at 2pm (sharp!)



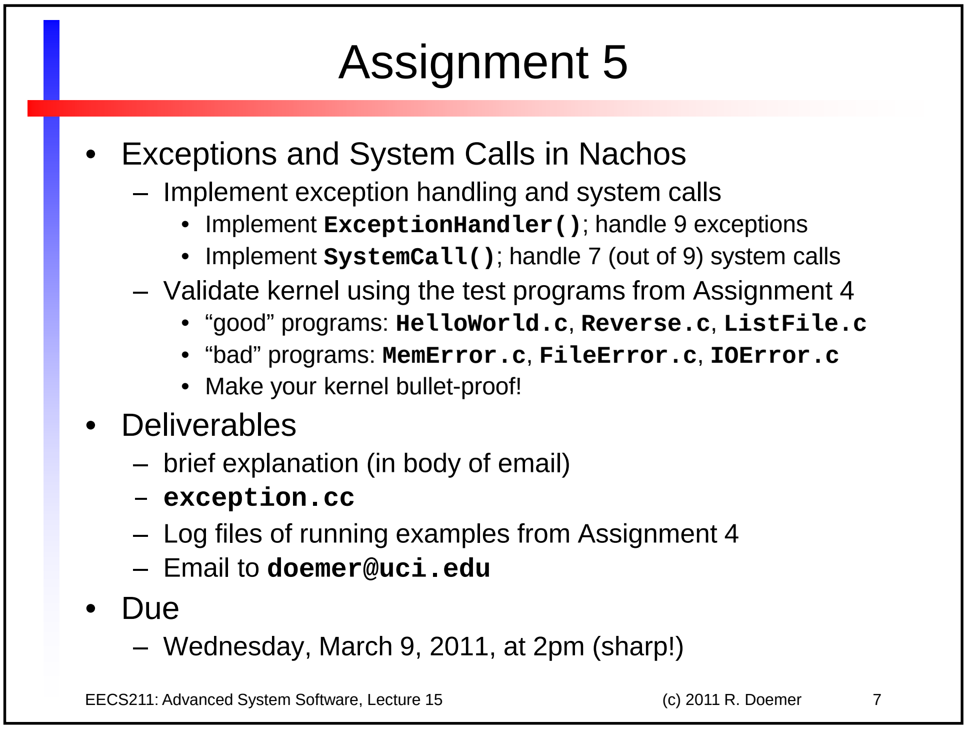
Assignments 4 and 5

- The Nachos System
  - User code:
    - Cross-compiled C/C++ code
    - emulated by MIPS simulator
  - Kernel:
    - Compiled C/C++ code
    - normal (debug'able) Unix process
  - I/O System:
    - simulated by Unix process I/O



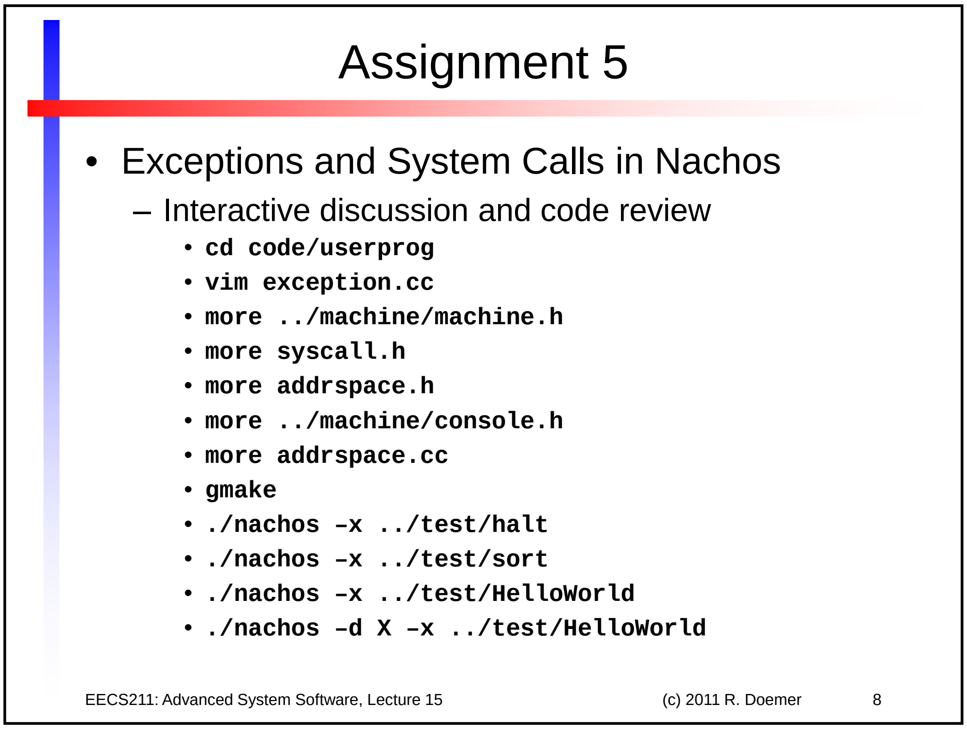
Assignment 4

- User programs in Nachos
  - Write simple user programs to be run on Nachos kernel
    - "good" programs: **`HelloWorld.c`**, **`Reverse.c`**, **`ListFile.c`**
    - "bad" programs: **`MemError.c`**, **`FileError.c`**, **`IOError.c`**
  - Validate kernel using these test programs
    - "good" programs should run successfully
    - "bad" programs should be caught and cleanly killed
- Deliverables
  - brief explanation (in body of email)
  - **`HelloWorld.c`**, **`Reverse.c`**, **`ListFile.c`**, **`MemError.c`**, **`FileError.c`**, **`IOError.c`**
  - corresponding log files
  - Email to **`doemer@uci.edu`**
- Due
  - Wednesday, March 2, 2011, at 2pm (sharp!)



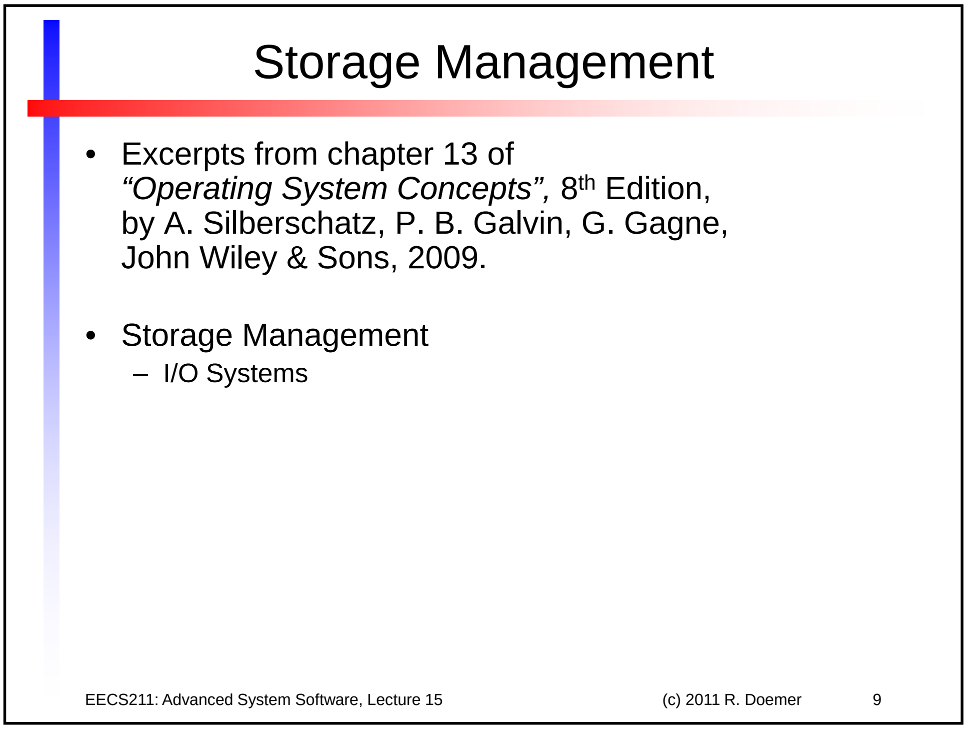
Assignment 5

- Exceptions and System Calls in Nachos
  - Implement exception handling and system calls
    - Implement **`ExceptionHandler()`**; handle 9 exceptions
    - Implement **`SystemCall()`**; handle 7 (out of 9) system calls
  - Validate kernel using the test programs from Assignment 4
    - “good” programs: **`HelloWorld.c`**, **`Reverse.c`**, **`ListFile.c`**
    - “bad” programs: **`MemError.c`**, **`FileError.c`**, **`IOError.c`**
    - Make your kernel bullet-proof!
- Deliverables
  - brief explanation (in body of email)
  - **`exception.cc`**
  - Log files of running examples from Assignment 4
  - Email to **`doemer@uci.edu`**
- Due
  - Wednesday, March 9, 2011, at 2pm (sharp!)



Assignment 5

- Exceptions and System Calls in Nachos
  - Interactive discussion and code review
    - **`cd code/userprog`**
    - **`vim exception.cc`**
    - **`more ../machine/machine.h`**
    - **`more syscall.h`**
    - **`more addrspace.h`**
    - **`more ../machine/console.h`**
    - **`more addrspace.cc`**
    - **`gmake`**
    - **`./nachos –x ../test/halt`**
    - **`./nachos –x ../test/sort`**
    - **`./nachos –x ../test/HelloWorld`**
    - **`./nachos –d X –x ../test/HelloWorld`**



Storage Management

- Excerpts from chapter 13 of
  “Operating System Concepts”, 8th Edition,
  by A. Silberschatz, P. B. Galvin, G. Gagne,
  John Wiley & Sons, 2009.

- Storage Management
  - I/O Systems