EECS 211: Advanced System Software Lecture 4

Rainer Dömer

doemer@uci.edu

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

Lecture 4: Overview

- The Nachos System
  - Introduction
  - Overview
- Assignment 1
  - Introduction to Nachos
- Assignment 2
  - Concurrency and Synchronization in Nachos

The Nachos System

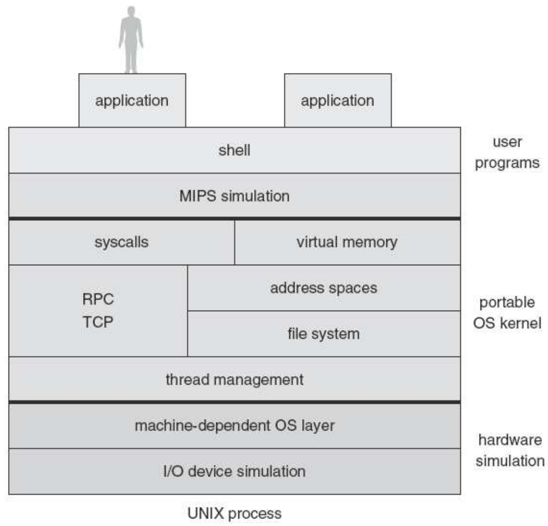
- Introduction
  - Instructional operating system
    - designed for teaching
  - Simple but working system
    - Concepts are learned by experimentation
  - Covers all major components of a modern OS
    - threads and process synchronization
    - file systems
    - multiprogramming
    - virtual memory
    - networking
  - Usable in normal Unix environment
  - Well-documented source code freely available

The Nachos System

- Documentation
  - Text book, Appendix D
  - Source code!
  - Additional resources
    - Nachos home page
    - Nachos roadmap
    - Wikipedia entry
- Why not Linux?
  - Size and complexity
  - System environment (“naked PCs”?)
  - Debugging (nightmare!)

The Nachos System

- Overview
  - User code: emulated by MIPS simulator
  - Kernel: normal (debug'able) Unix process
  - I/O System: simulated by std. process I/O



Assignment 1

- Schedule
  - posted Jan 16, 2008 (week 2)
  - due Jan 23, 2008 (week 3)
- The Nachos System
  - Task 1: Read the overview chapter
    - Text book, Appendix D (contents online)
  - Task 2: Setup the software
    - Setup environment, copy tar-ball, unpack, compile, test
  - Task 3: *Understand* the Nachos system!
    - Read documents and source code
- Deliverable
  - log output when running the plain Nachos installation
  - Email to doemer@uci.edu

Assignment 2

- Schedule
  - posted Jan 16, 2008 (week 2)
  - due Jan 30, 2008 (week 4)
- The Nachos System
  - Task 1: Analyze the thread mechanism
    - file **`threads/threadtest.cc`** and others
  - Task 2: Implement locks and condition variables
    - files **`synch.h`** and **`synch.cc`**
- Deliverables
  - brief explanation of **`Yield`** and **`SWITCH`** functions
  - code for locks and condition variables
  - test run of using condition variables for alternating threads
  - Email to doemer@uci.edu