Lecture 1
Introduction

Tevfik Koşar
Louisiana State University
January 15th, 2007

Contact Information

- Instructor: Prof. Tevfik Kosar
  - Office: 292 Coates (also 333 Johnston)
  - Phone: 578-9483
  - Email: kosar@csc.lsu.edu
  - Web: http://www.cct.lsu.edu/~kosar
  - Office hours: Tue & Wed, 11:00am – noon (Or anytime by appointment)

- Teaching Assistant: TBD

Logistics

- Course web page: http://www.cct.lsu.edu/~kosar/csc4103
  - All lecture notes will be available online
  - As well as homework assignments, projects and other important course information

- Course mailing list: csc4103@cct.lsu.edu
  - Important course announcements including projects, homework assignments, and exams will be sent to this mailing list
  - Provide me with your active email address to be added to the class mailing list

Textbooks

- Operating System Concepts
- Introduction to Operating System Design and Implementation

Grading

- The end-of-semester grades will be composed of:
  - Pop Quizzes: 10%
  - Homework: 15%
  - Projects: 20%
  - Midterm: 25%
  - Final: 30%

You are expected to attend the classes and actively contribute via asking and/or answering questions.

Rules

- No late homework/project submissions accepted!
- No computers/laptops will be allowed in regular class as well as exam.
- Exams will be closed book.
- You are only responsible from material covered in the class, homework, and projects.
- Academic dishonesty will be treated seriously.
What Expect to Learn?

- Basic Concepts of Operating Systems
- Operation, Resource Utilization, Management
- Processes, Threads and Concurrency
- CPU and I/O Scheduling
- Memory and Storage Management
- File System Structures
- Synchronization and Deadlocks
- Protection and Security
- Distributed Computing & Related Issues

What is an Operating System?

- A program that manages the computer hardware.
- An intermediary between the computer user and the computer hardware.
- Manages hardware and software resources of a computer.

Role of an Operating System

- "Pyramid" View - Silberschatz
- "Layered" View - Tanenbaum
- "Layered & stairs" View - Stallings
Role of an Operating System

In Short

• An operating system is a program that acts as an intermediary between user (applications) and the computer hardware.

Operating System Goals

• From the user perspective
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use

• From the System/HW Perspective
  - Manage the resources
  - Use the computer hardware in an efficient manner

OS Services for Users

• Program Execution
  - The OS loads programs and data into memory, initializes I/O devices and files, schedules the execution of programs

• Access to I/O Devices
  - The OS hides I/O device details from applications (direct I/O access is forbidden) and offers a simplified I/O interface

• Controlled Access to Files
  - The OS organizes data into files, controls access to the files (i.e. create, delete, read, write) and preserves their integrity

Communications

• The OS allows exchange of information between processes, which are possibly executing on different computers

Error Detection and Response

• The OS properly handles HW failures and SW errors with the least impact to running applications (i.e. terminating, retrying, or reporting)
OS Services for System/HW

- Resource Allocation
- Operation Control
- System Access
- Accounting and Usage Statistics

Operation Control
- The OS controls the execution of user programs and operations of I/O devices

System Access
- The OS ensures that all access to resources is protected, including authorization, conflict resolution etc.

Accounting and Usage Statistics
- The OS keeps performance monitoring data

Computer System Organization

- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles

Computer-System Operation

- I/O devices and the CPU can execute concurrently.
- Each device controller is in charge of a particular device type.
- Each device controller has a local buffer.
- CPU moves data from/to main memory to/from local buffers
  - If no CPU involved \(\rightarrow\) DMA
- I/O is from the device to local buffer of controller.
- Device controller informs CPU that it has finished its operation by causing an interrupt.

Common Functions of Interrupts

- Interrupt transfers control to the interrupt service routine generally, through the interrupt vector, which contains the addresses of all the service routines (interrupt handlers).
- Interrupt architecture must save the address of the interrupted instruction. (also save state of CPU, eg. registers, PC)
- Incoming interrupts are disabled while another interrupt is being processed to prevent a lost interrupt.
- A trap is a software-generated interrupt caused either by an error or a user request.
- An operating system is interrupt driven.

Interrupt Timeline for I/O
I/O Structure

- After I/O starts, control returns to user program only upon I/O completion – **synchronous**
  - Wait instruction idles the CPU until the next interrupt
  - Wait loop (contention for memory access).
  - At most one I/O request is outstanding at a time, no simultaneous I/O processing.
- After I/O starts, control returns to user program without waiting for I/O completion – **asynchronous**
  - System call – request to the operating system to allow user to wait for I/O completion.
  - Device-status table contains entry for each I/O device.

Two I/O Methods

![Diagram of I/O Methods](image)

Operating System Operations

- **Interrupt driven** by hardware
- Unexpected errors can happen anytime
  - Software error or request creates **exception** or trap
    - e.g., division by zero, invalid memory access
  - Other process problems include infinite loop, processes modifying each other or the operating system
- **OS needs to protects itself**
  - **Dual-mode operation**

Dual-Mode Operation

- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
    - Mode bit provided by hardware
      - Provides ability to distinguish when system is running user code or kernel code
      - Protects OS from errant users, and errant users from each other
      - Some instructions designated as **privileged**, only executable in kernel mode
      - System call changes mode to kernel, return from call resets it to user

Transition from User to Kernel Mode

- How to prevent user program getting stuck in an infinite loop / process hogging resources
  - **Timer**: Set interrupt after specific period (1ms to 1sec)
    - Operating system decrements counter
    - When counter zero generate an interrupt
    - Set up before scheduling process to regain control or terminate program that exceeds allotted time

Summary

- **What is an OS?**
- **Operating System Goals**
  - User View vs System View
- **Operating System Services**
- **Computer System Operation**
  - Interrupts
- **Synchronous vs Asynchronous I/O**
- **User Mode vs Kernel Mode**

**Reading Assignment**: Chapter 1 from Silberschatz.
Acknowledgements

- “Modern Operating Systems” book and supplementary material by A. Tanenbaum
- R. Doursat and M. Yuksel from UNR