CS202 – Advanced Operating Systems

Introduction

January 6, 2025

What is this course about?

- How has the role of the operating system evolved over time?
  - How does the past inform the present?
- What are the principles that underlie Operating Systems?
- What are current and future trends in OS?
- Make it real: projects to get some experience with OS development and research
- *Get you ready to do Systems Research!*

A little bit about me

- Have worked on OS projects
  - L4 microkernel research
  - Security liaison to the IBM Linux Technology Center
    - Linux Security Modules (LSM) and SELinux
    - Linux Integrity Measurement Architecture (IMA)
  - In Academia – mostly security focused
    - Access control, File systems security, Driver security, TEEs, Kernel integrity
- But, I don’t think of myself as an OS person
  - My favorite two answers are
    - I don’t know
    - What do YOU think?
  - I am looking forward to learning with you

Class format

- For every topic:
  - Some review (undergrad OS) and research
  - Discuss research papers
- Research papers:
  - Review required for some papers (7 papers during the quarter)
    - Additional papers discussed in class
    - You are responsible for required papers and material discussed in class
    - Discuss review format next time

Reading Research Papers

- Guidelines for reading papers
  - Make sure to identify authors' goals and assumptions. Not always directly stated.
  - Look for high-level takeaways.
  - Follow a multi-pass reading strategy
    - Pass1: Get overview. Pass2: Read details and make notes. Pass3: Re-read details to evaluate.
  - Think how techniques used are applicable today. Identify extensions.
- Specifics on review format next time

Course Logistics

□ Projects	← 20%
□ Reading and critiquing papers	} 20%
□ Quizzes	
□ Exams	← 55%
□ Participation	← 5%

Projects

- Projects (3)
  - Programming on xv6
  - Or alternative of a research project for P3
- Teams of two
  - Same team throughout the quarter
- Often not directly connected to the research topics we discuss
  - Primary goal is to improve systems-building skills

Quizzes

- Will be four quizzes during the quarter
  - Mainly to test review material
  - 3-4 questions per quiz

- Goal is to achieve 10pts for the quarter to gain full credit for the 10% quiz grade.
  - Thus, 100% not necessary to get 100% of credit
  - Extra credit above that

Exams

- One take-home midterm – 2/12 to 2/14
  - Review questions
  - Open-ended questions
  - About 50/50

- Final exam is in-class and cumulative
  - Again, will have review questions and research questions

Course Material

- Will provide a review of undergraduate material (probably not enough)

  - For more background:
    - Three Easy Pieces: http://pages.cs.wisc.edu/~remzi/OSTEP/
      - Its free!
      - Its excellent!

- **Significant material from research papers**

Prerequisites

- Will recap basics of OS, but it will be quick
- **To do well, you must acquire undergrad OS or equivalent preparation**
- **Architecture, security, distributed systems courses are also a plus.**

Resources

- Syllabus and Course Schedule will be posted incrementally on course website
  - Schedule will evolve during the quarter – check often

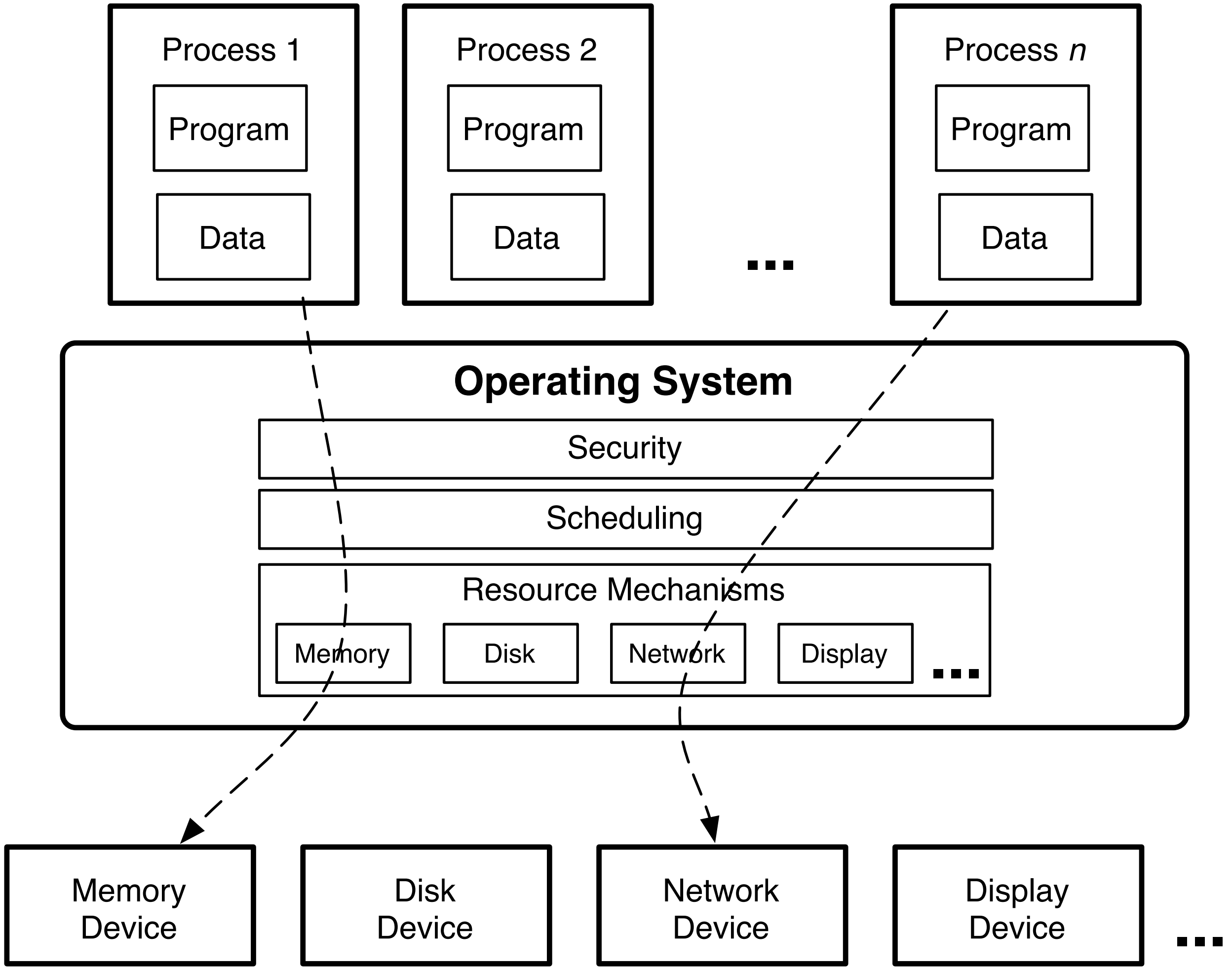
http://www.cs.ucr.edu/~trentj/cs202-w25/

- Canvas
  - I will make announcements
  - Prefer you email me at trentj@ucr.edu
  - Nurlan is the TA – contact at nurlan.nazaraliyev@email.ucr.edu
- Slack channel

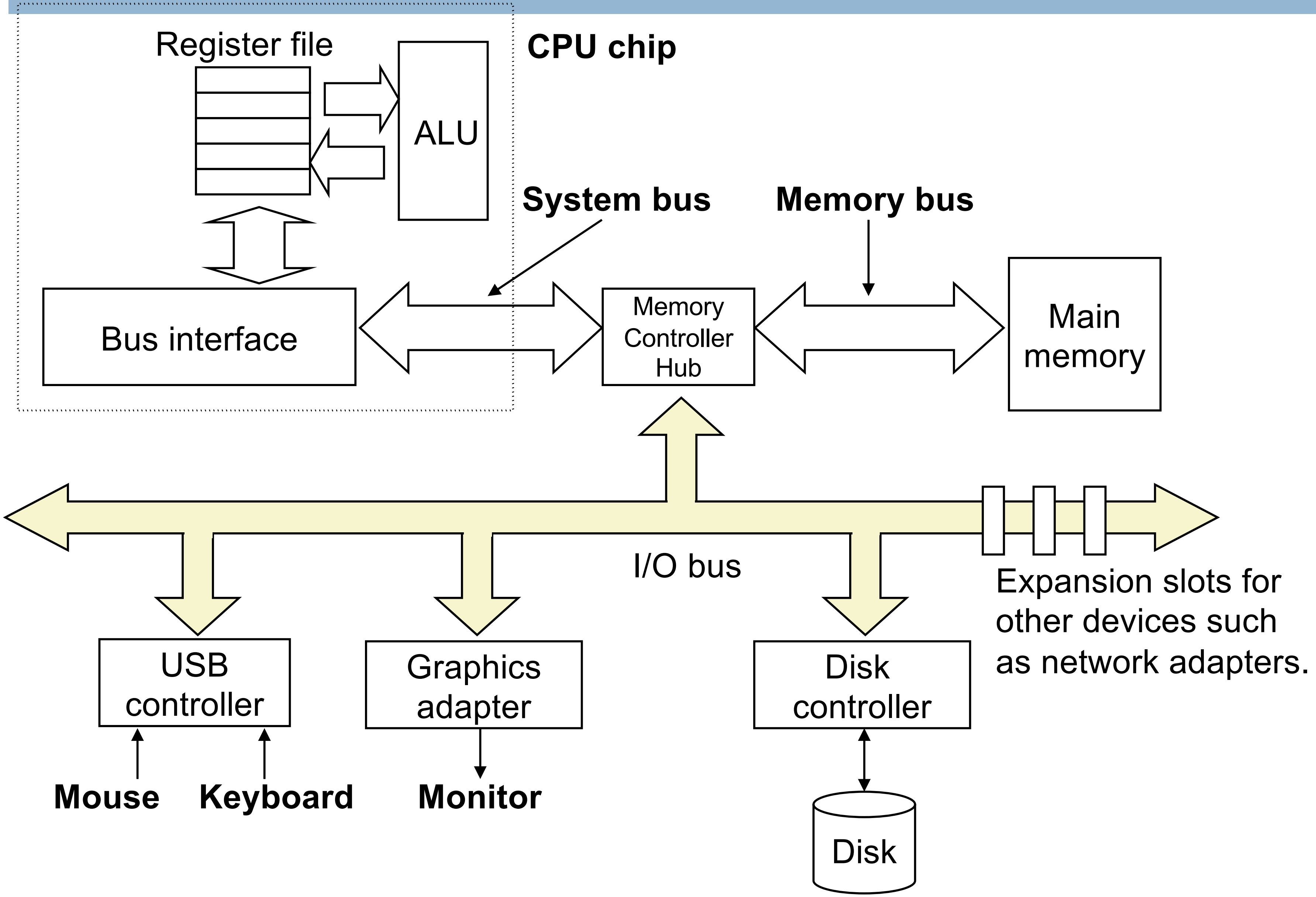
OS Is Cool!

- May be hard to believe, but it is true
- Covers all facets of computer science
  - Architecture, software, algorithms
  - Something for everyone!
- An extremely valuable skill set
  - If you can do systems programming well, you can get a job anywhere
- At the core of computer science

Situation


Process 1
Program
Data
Process 2
Program
Data
...
Process *n*
Program
Data
Operating System
Security
Scheduling
Resource Mechanisms
Memory
Disk
Network
Display
...
Memory Device
Disk Device
Network Device
Display Device
...

OS Abstracts Hardware



Getting more technical

- What is an OS?
  - A piece of software that *virtualizes* hardware and *arbitrates* it
- A manager in a shop
  - Directs resources
    - Controls CPUs, memory, devices…
  - Enforces working policies
    - Fairness, resource limits, security, …
  - Simplifies complex tasks
    - Abstracts hardware; offers system calls

Abstraction

- Programmers do not want to deal with hardware directly
  - Lots of complex hardware
  - Multiple hardware devices do the same thing
    - Network devices, storage devices, memory, etc.
- The operating system provides abstractions of these hardware resources that you work with
  - Processes, files, sockets, etc.
  - Tries to hide the complexity from programmers
    - E.g., virtual memory

Algorithms and Policy

- Need to provide ways for multiple processes / threads to share hardware resources effectively
  - Efficiently, security, fairly, …
- But, such problems are often computationally complex
  - NP-complete, such as scheduling
  - So, need heuristic algorithms – implementing policies
- Regarding policies
  - Separate policy from algorithms
  - Select good policies – continually evolving

Three main themes

- Virtualization
  - Create models to simplify use of hardware resources
  - To allow them to be shared
    - Scheduling and security
- Concurrency
  - Support many activities to work with limited resources at the same time
- Persistence
  - I/O and File systems

Some topics we will cover

- Operating Systems models and how they evolved
  - Monolithic kernels, micro-kernels, …
    - extensibility, scalability, security, …
  - How do these models influence current OS organizations
    - Modularity, virtualization, containers, …
- Concurrency:
  - Synchronization
  - Multicore OS
- File systems:
  - Sequential, networked, distributed, internet scale
- How do they evolve to new environments…
  - Multicore, Distributed systems, …

Catering to Applications

- Provide resource needs of an application
  - CPU, memory, device access
- When applications launch, the OS loads the program from file into memory
  - Allocates memory for code, data, heap and stack
  - Can the application ask for more resources?
    - Yes, the OS receives additional requests and provides resources as needed
- OS also reacts to events in the system
- Gets out of the way as fast as possible

Processor virtualization

- Abstractions
  - Program: static entity
  - Process: program in execution
    - Unit of resource allocation and one thread of execution
  - Threads
    - More than one thread of execution in one context
- Schedule and secure their execution

Memory management

- Abstractions:
  - Address space for each processor

- OS implements these abstractions using the available hardware support
  - Paging, segmentation, TLBs, caches…

- What happens in emerging systems?
  - Disaggregation; RDMA; Persistent memory...

Storage/file system

- Abstraction: Files and directories
  - Implemented in a variety of different ways
- Traditional file system: mapping of files to storage
    - Disk vs. flash vs. PCMs
  - Network file system
  - Distributed FS
  - Internet scale FS

Conclusions

- Today was a quick overview of the role of an OS
- Goal is to get you ready to discuss OS organization and evolution, our first topic
- Quick overview of the history of OSes next time
  - Skim the Thompson article
  - We will discuss what is expected of paper reviews/critiques next time in the context of this
- Then processes and memory
  - Please read on your own if you need to refresh
    - From the Three Easy Steps textbook

Questions