



# Why Grad OS?



- ✦ Primary Goal: Demystify how computers work

# An example progression



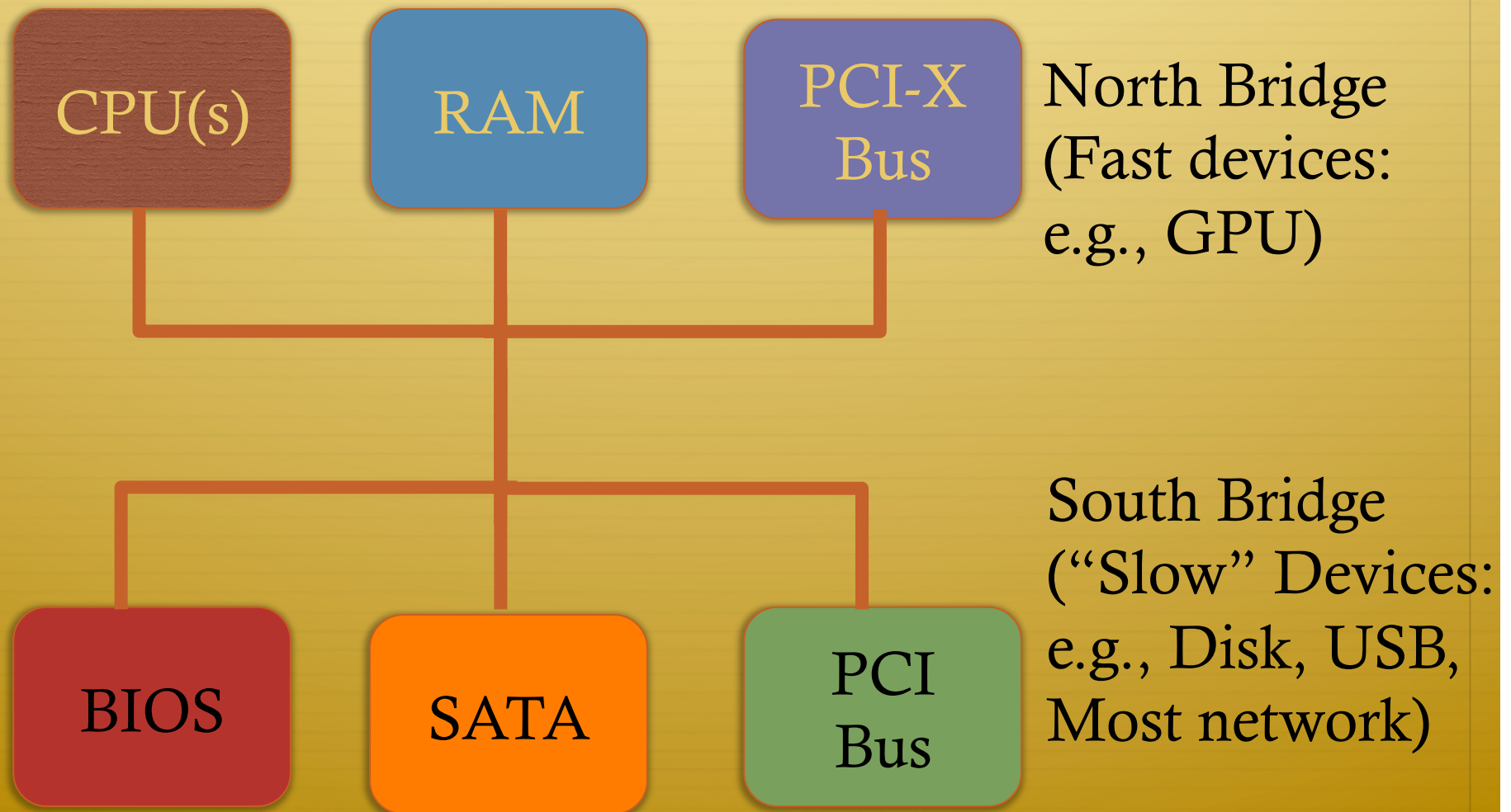
- ✦ Undergrad OS:
  - ✦ High-level understanding of paging
  - ✦ Theoretical issues like fragmentation
- ✦ Grad OS (506): Build a pager
  - ✦ Solid understanding of how paging SW + HW work
- ✦ Advanced Grad OS (624): Read novel research papers
  - ✦ Do creative things with paging: virtualization, security, etc

# 506: Learn by doing



- ✦ You will write major chunks of your own OS
  - ✦ Memory management, context switching, scheduler, file system, IPC, network driver, shell, etc.
  - ✦ Linux scheduler:
    - ✦ Difficult to understand just by reading source
    - ✦ Small modifications require first understanding the code
    - ✦ Impossible to replace/reimplement
  - ✦ No substitute for building it yourself!

# A logical view of hardware



# Fewer Bridges



- ✦ Newer system organizations are moving more devices to the North bridge, and consolidating more things on the CPU itself.

# A logical view of the OS

Binary  
Formats

Memory  
Allocators

Threads

User

System Calls

Kernel

RCU

File System

Networking

Sync

Memory  
Management

Device  
Drivers

CPU  
Scheduler

Interrupts

Disk

Net

Consistency

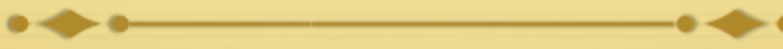
Hardware

# JOS

- ✦ Developed at MIT, used at several top schools
  - ✦ The “J” is for Josh Cates, not Java
- ✦ In C and Assembly, boots on real PC hardware
  - ✦ You get the skeleton code, fill in interesting pieces
- ✦ Build the right intuitions about real OSes
  - ✦ but with much simpler code

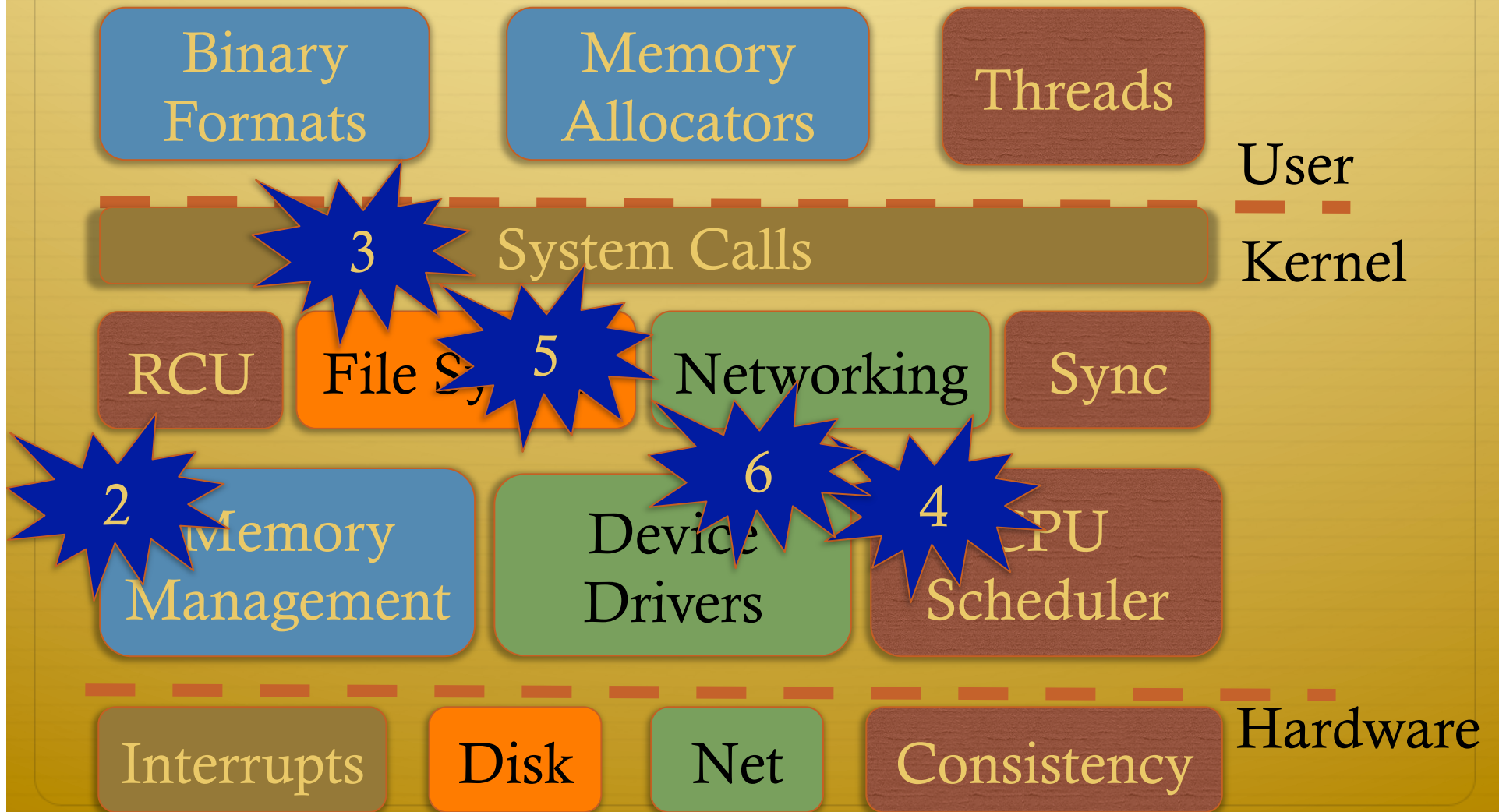


# Labs, cont.



- ✦ This course is **coding intensive**
  - ✦ You should know C, or be prepared to remediate quickly
  - ✦ You will learn basic, inline x86 assembly
  - ✦ You must learn on your own/with lab partner
- ✦ The lab is difficult, but worthwhile
  - ✦ You will want to commemorate, with a T-shirt, tattoo, etc.

# JOS Labs



# Last Lab



- ✦ Includes open ended project
  - ✦ Can add significant feature to JOS
  - ✦ Or do a research task on another system
- ✦ Plan ahead – proposals due 10/23
  - ✦ Note all deadlines on course website

# Challenge Problems



- ✦ Each lab includes challenge problems, which you may complete for bonus points (generally 5—10 points out of 100)
  - ✦ Unwise to turn in a lab late to do challenge problems
  - ✦ Can complete challenge problems at any point in the semester---even on old labs
- ✦ Indicate any challenge problems completed in challenge.txt file

# CSE 522



- ✦ This course can also count as your MS project course (CSE 522)
- ✦ Requirements: Same as 506, except:
  - ✦ You must do the labs alone
  - ✦ You must complete 1 challenge problem in each lab
- ✦ To enroll: you must first be in 506
  - ✦ Ask me and I will have you moved to 522

# No Textbook



- ✦ You're welcome
- ✦ Several recommended texts
  - ✦ Several free on SBU safari online site
  - ✦ Others on reserve at library
  - ✦ Required readings will mainly be papers you can print out

# Lectures



- ✦ Compare and contrast JOS with real-world OSes
  - ✦ Mostly Linux, some Windows
- ✦ Supplement background on hardware programming
  - ✦ Common educational gap between OS and architecture

# SBU Capture



- ✦ Experiment: TLT will be recording the projection and audio (no video of me, sadly)
  - ✦ Recordings will be automatically posted to BlackBoard
  - ✦ Intended to help you study
- ✦ **This is best effort**
  - ✦ No guarantee all lectures will be recorded
- ✦ **This is no substitute for lecture attendance**
  - ✦ Can't ask questions
- ✦ **If attendance suffers, I will stop recording lectures**



# Prerequisites



- ✦ Undergrad OS
  - ✦ In some cases, industry experience is ok
  - ✦ Worth brushing up if it has been a while
  - ✦ **In-class quiz**, due before you leave
    - ✦ **If you can't answer 50% of these questions, consider ugrad OS**
- ✦ C programming
- ✦ Basic Unix command-line proficiency
- ✦ See me if you have already done the JOS lab, or similar

# Space in the class



- ✦ Wait list is currently full
- ✦ Grad students often over-enroll
  - ✦ Space likely to open up in first week
  - ✦ If you want in, keep showing up for a few lectures
- ✦ Worst case: Prof. Zadok teaching 506 in spring
  - ✦ Likely to be offered every semester going forward

# Course email list



- ✦ Sign up at <http://lists.cs.stonybrook.edu/mailman/listinfo/cse506>
- ✦ This is the primary announcement medium
- ✦ And for discussions about course work
  - ✦ Do not post code here or other solutions
  - ✦ Goal: Everyone can learn from general questions
- ✦ Material discussed on the mailing list can be an exam question

# Other administrative notes



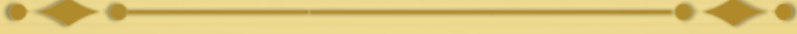
- ✦ Read syllabus completely
- ✦ Subscribe to the class mailing list
- ✦ 2 exams cover: lectures, labs, mailing list
- ✦ Every student will get a VM for lab work
  - ✦ You may use your own computer, staff can't support it
- ✦ All staff email goes to [cs506ta@cs.stonybrook.edu](mailto:cs506ta@cs.stonybrook.edu)
  - ✦ Except private issues for instructor only

# VM Assignments



- ✦ Your VM is cse506-USER, where USER is your netid
- ✦ Each VM is hosted on the server esx1sc---esx4sc
  - ✦ You should receive an email with your server and initial password
- ✦ The account is cse506
- ✦ Once it is powered on, it will listen for ssh on port 130
- ✦ **Change the password immediately**

# Lab Partners



- ✦ Can work alone, but better with help
  - ✦ Some excellent students earned A's working alone
  - ✦ Many good students earned B's working alone
  - ✦ No need to be a hero
- ✦ Choose your own partners
  - ✦ Lab mailing list good for finding them
- ✦ Same for entire course
  - ✦ Changes only with instructor permission

# To Do



- ✦ Email me your partner selection
- ✦ We will then create the git repository you will use to turn in your assignments
- ✦ In the meantime, clone the read-only, http repository to get started
- ✦ Please do this well in advance of the deadline

# Academic Integrity



- ✦ I take cheating very seriously. It can end your career.
- ✦ In a gray area, it is your job to stay on right side of line
- ✦ Never show your code to anyone except your partner and course staff
- ✦ Never look at anyone else's code (incl. other universities)
- ✦ Do not discuss code; do not debug each other's code
- ✦ Acknowledge students that give you good ideas



# Lateness



- ✦ Each group gets 72 late hours
  - ✦ List how many you use in slack.txt
  - ✦ Each day after these are gone costs a full letter grade on the assignment
- ✦ It is your responsibility to use these to manage:
  - ✦ Holidays, weddings, research deadlines, conference travel, Buffy marathons, release of the next Zelda game, etc.
- ✦ 3 Exceptions: illness (need doctor's note), death in immediate family, accommodation for disability

# Lab 1 assigned



- ✦ Due Friday, 9/7 at 11:59 pm, eastern.
- ✦ Instructions on website
- ✦ Quick demo

# Getting help



- ✦ TA's (TBD) will keep office hours
- ✦ Instructor keeps office hours
  - ✦ Note that “by appointment” means more time available on demand

# Questions?

