LECTURE 1: INTRODUCTION

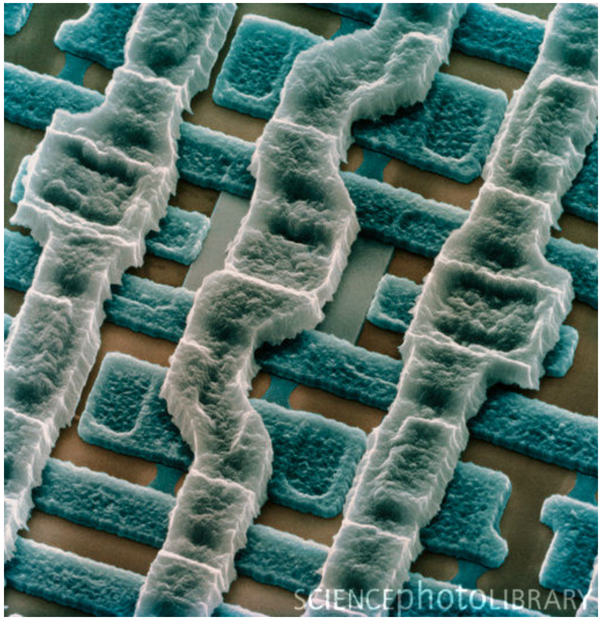
Computer Systems and Networks

Dr. Pallipuram
(vpallipuramkrishnamani@pacific.edu)

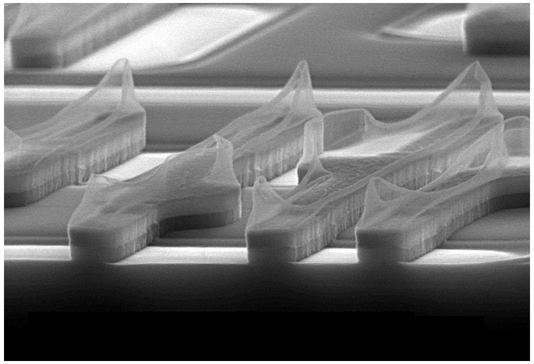
Our Course Goals

Bottom Up Picture

Memory cell

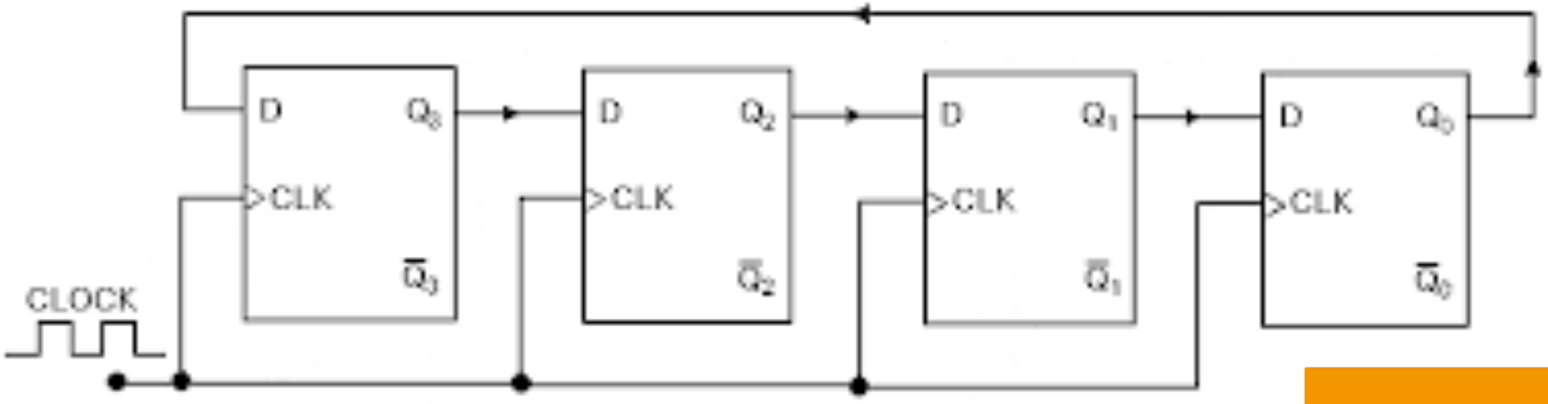


Transistor



Level 0: Electronics and Circuits (ECPE 41, 131)

Bottom Up Picture



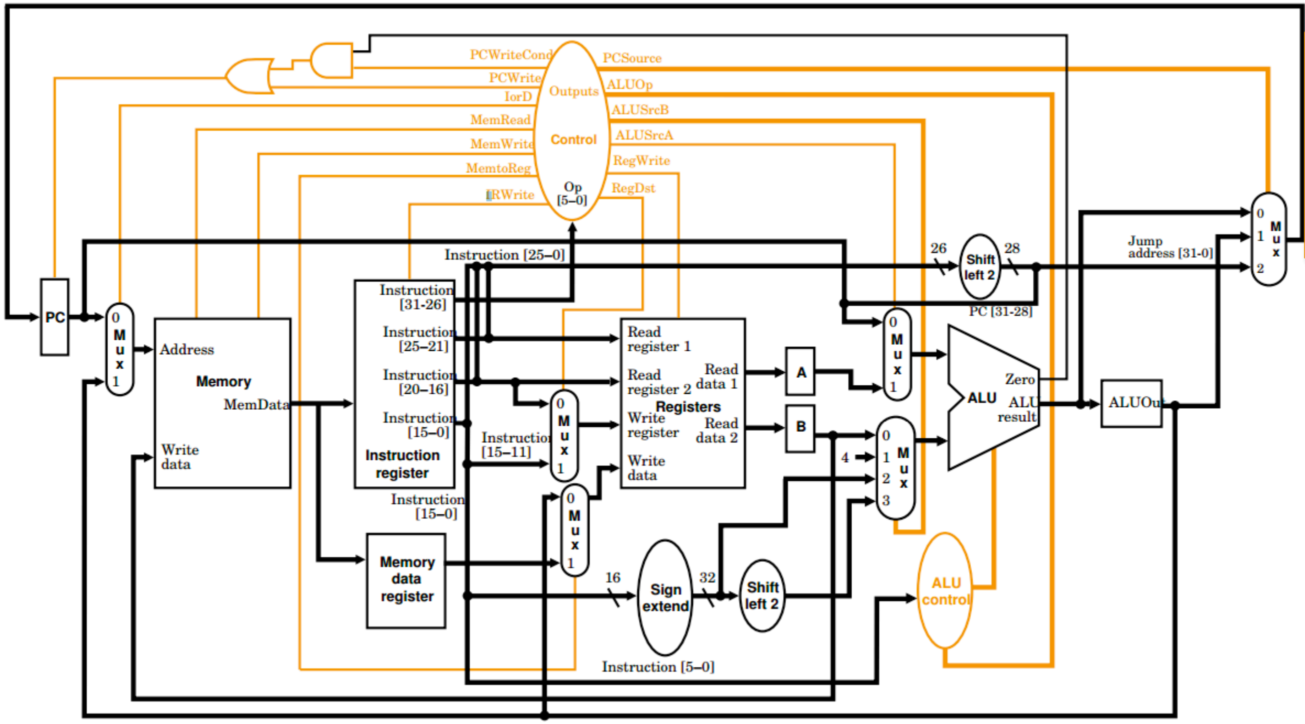
what does this circuit do? write first few sequences starting from 0000

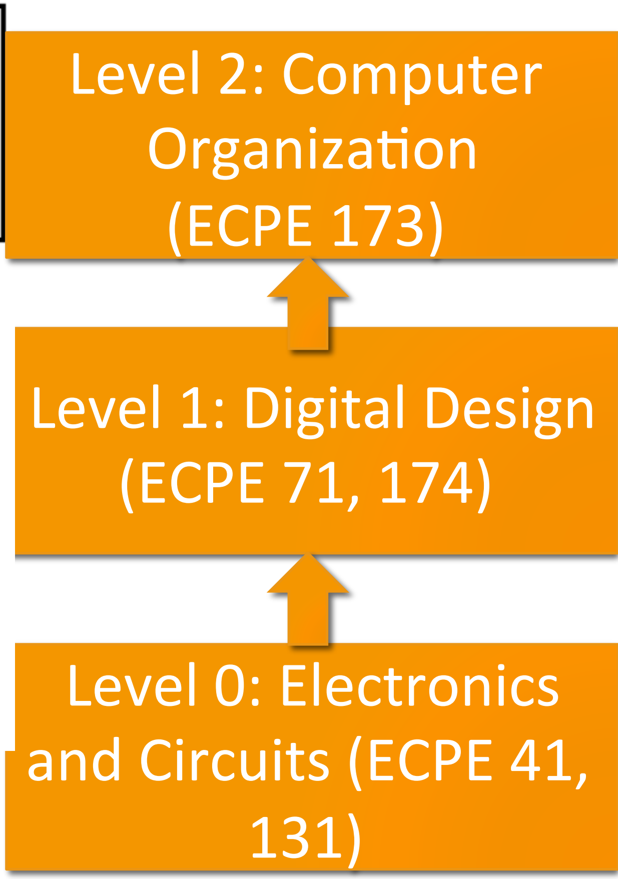
Level 1: Digital Design (ECPE 71, 174)

Level 0: Electronics and Circuits (ECPE 41, 131)

Bottom Up Picture

How about this one?

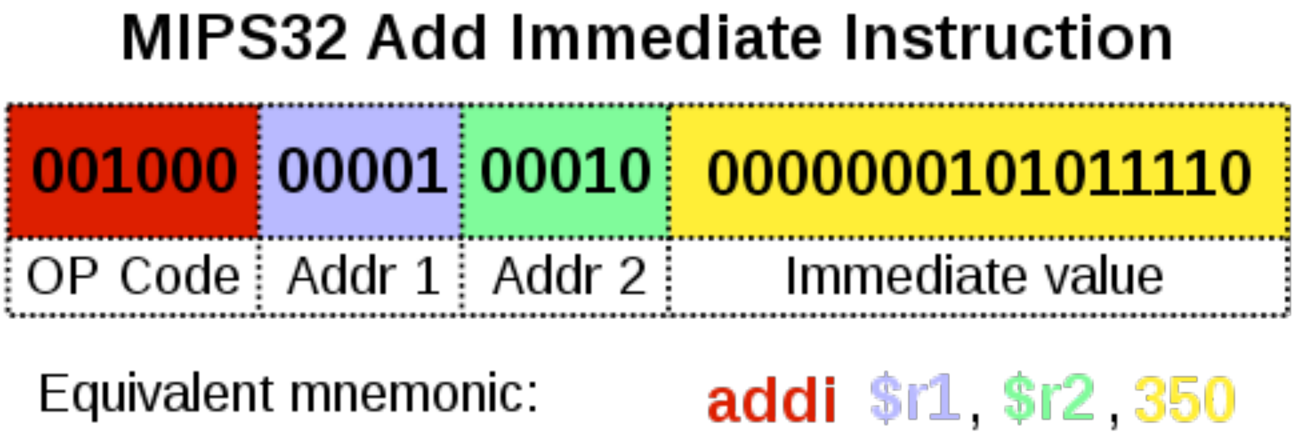




Level 2: Computer Organization (ECPE 173)

Level 1: Digital Design (ECPE 71, 174)

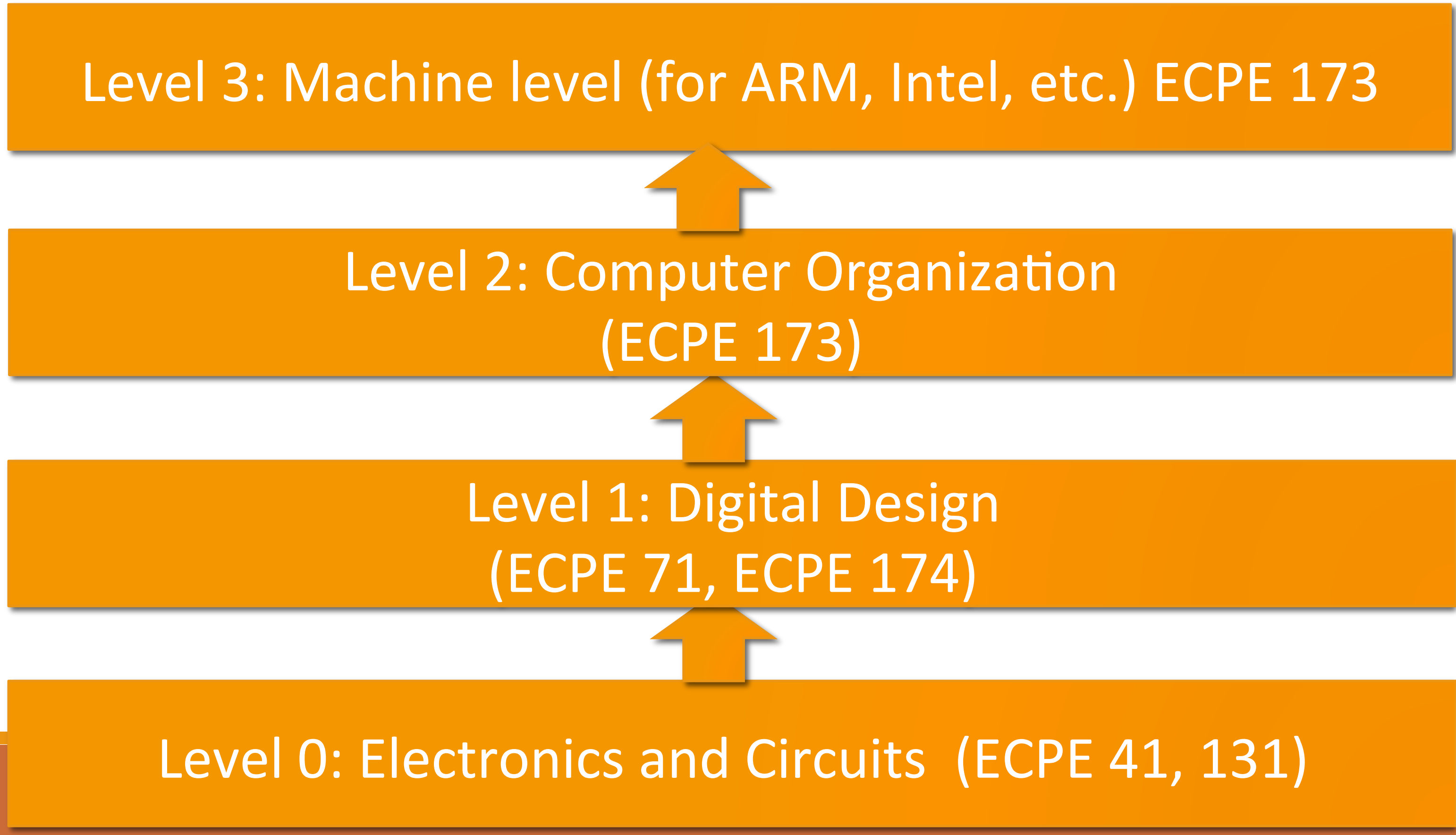
Level 0: Electronics and Circuits (ECPE 41, 131)



MIPS32 Add Immediate Instruction

001000	00001	00010	0000000101011110
OP Code	Addr 1	Addr 2	Immediate value

Equivalent mnemonic: addi $r1, $r2, 350

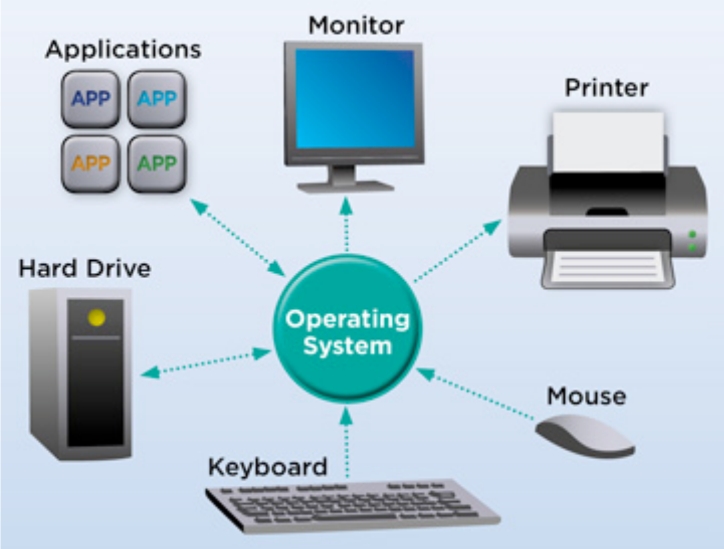


Level 3: Machine level (for ARM, Intel, etc.) ECPE 173

Level 2: Computer Organization (ECPE 173)

Level 1: Digital Design (ECPE 71, ECPE 174)

Level 0: Electronics and Circuits (ECPE 41, 131)



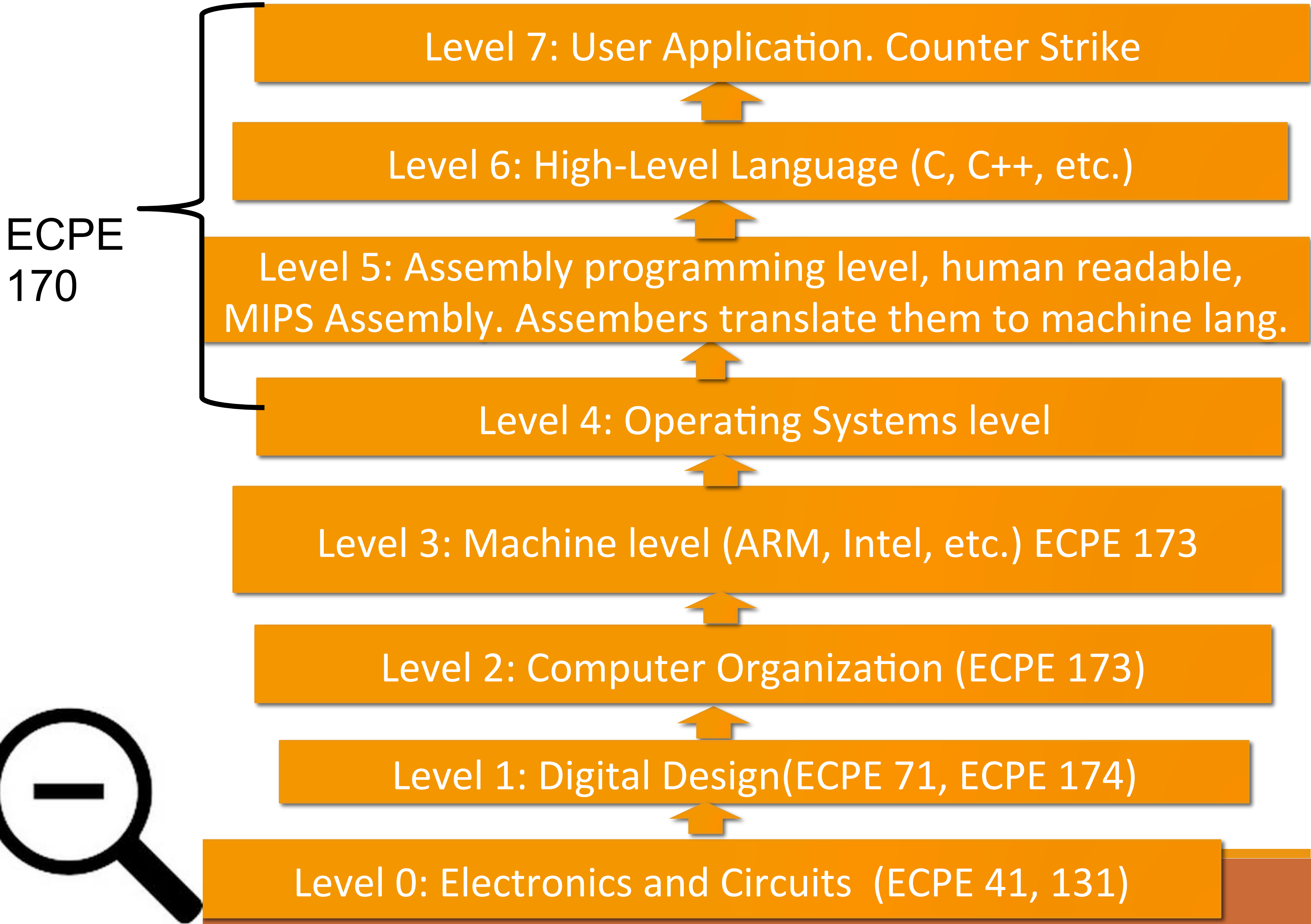
l 4: Operating Systems level

Level 3: Machine level (ARM, Intel, etc.) ECPE 173

Level 2: Computer Organization
(ECPE 173)

Level 1: Digital Design
(ECPE 71, ECPE 174)

Level 0: Electronics and Circuits (ECPE 41, 131)



ECPE 170

Level 7: User Application. Counter Strike

Level 6: High-Level Language (C, C++, etc.)

Level 5: Assembly programming level, human readable, MIPS Assembly. Assembers translate them to machine lang.

Level 4: Operating Systems level

Level 3: Machine level (ARM, Intel, etc.) ECPE 173

Level 2: Computer Organization (ECPE 173)

Level 1: Digital Design(ECPE 71, ECPE 174)

Level 0: Electronics and Circuits (ECPE 41, 131)

Course Goals

What do you, as a programmer, need to know about the underlying system (software *and* hardware) to write more efficient code?

- Role of the tools
  - Compiler, assembler, linker, profiler
- Role of the operating system and its efficient usage
- Assembly programming (using the CPU efficiently)
- Memory hierarchy and its impact on performance

C Programming Language

C

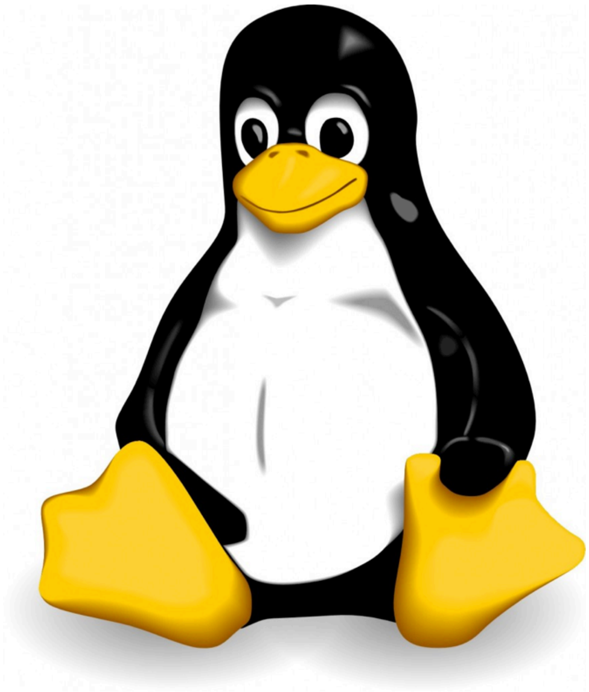
Why not Python, Java, Ruby, Perl, PHP, …?

High-level languages (especially interpreted, managed code…) try to *hide* the underlying machine from you

ECPE 170 wants to *reveal* the underlying machine to you!

Bay area companies want more C

Linux



Course will be taught 100% in Linux

Feedback from co-op employers and graduates: “More Linux/Unix skills please!”

Software companies work with some version of Unix. Why?

Virtual Machine

Course will be taught 100% from a virtual machine booting Linux that *you* install!

Couldn't you just give us remote access to a server someplace that is already configured?

Yes, but…

- By installing it yourself you will have the skills to use it again in the future
- No mysterious "Professor Pallipuram" software configuration

Version Control

Course will use version control!

- Only way to get lab code or turn in assignments

Did you have *to mandate VCS for ECPE 170?*

No, not really, but…

- Too many Pacific graduates were *avoiding* learning this on their own!
- **Feedback from co-op employers and graduates: "Only n00bs work without version control!"**
- Used everywhere: Source code of all kinds! (C++, Python, Matlab, VHDL/Verilog, …)

Course Mechanics

Websites

Main website (syllabus, schedule)

- http://ecs-network.serv.pacific.edu/ecpe-170

Canvas website (gradebook)

- http://canvas.pacific.edu

Bitbucket.org (version control)

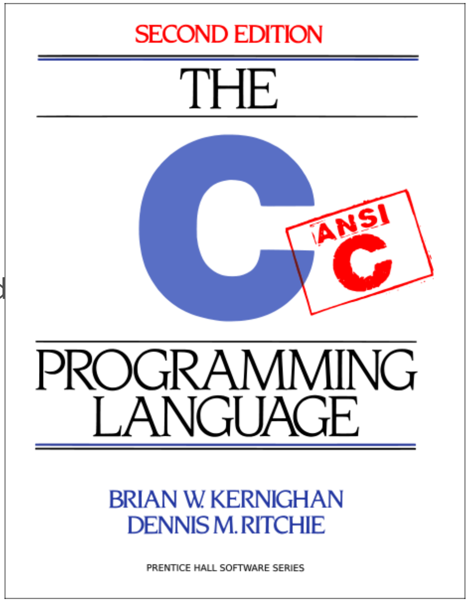
- http://bitbucket.org

Textbook

No official textbook

Optional reference books
(useful for this class and beyond)

- The C Programming Language, 2nd Edition
- See MIPS Run, 2nd Edition



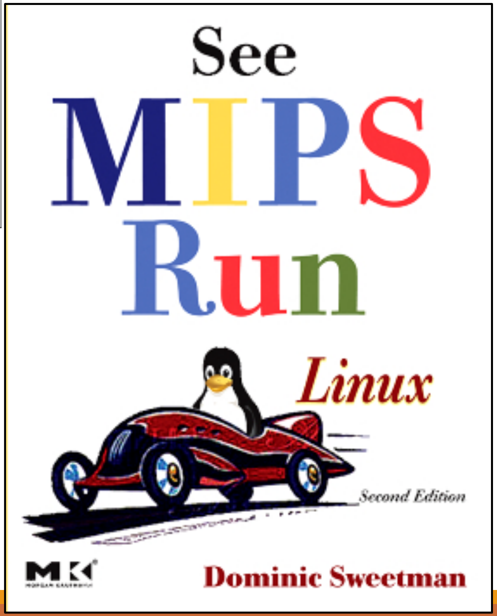
SECOND EDITION

THE C PROGRAMMING LANGUAGE

ANSI C

BRIAN W. KERNIGHAN
DENNIS M. RITCHIE

PRENTICE HALL SOFTWARE SERIES



See MIPS Run

Linux

Second Edition

Dominic Sweetman

Grading

30% - Exams

- 15% - Mid-term exam
- 15% - Final exam

65% - Labs

- Points assigned to each lab will vary based on complexity
- Each lab *begins* as an in-class activity
  - Unfinished work becomes homework/project
  - **Labs are large – assume “the usual” amount of homework/projects for a 4-credit class**
- **Tip: The best students last semester *started* the labs outside of class, and finished them as an in-class activity**
- **L2% penalty per late day. No more than 7 late days**

5% - In-class participation

Class Attendance

Strong attendance requirement may entail significant portion covered in the lecture and/or in-class participation problem

Moderate attendance requirement entails significant lab activity

Must be present for 8 AM-8:55 AM or 9 AM – 9:55 AM

Skip no more than two consecutive moderate classes

Recommended attendance requirement entails that students have the option of performing their work outside of the class, as long as they are confident in performing the required task on their own.

Honor Code

All assignments are submitted individually

Encouraged Activities

- Collaborating with your classmates (asking questions, solving problems together)
- Searching for solutions online
  - Provided code copied does not exceed 25% of total assignment length
  - Provided you clearly **document this copy** in your source code and lab report
    - What did you copy? Where did it come from?

Honor Code

Risky Activities

- Having your classmates type on your computer or assignment file

Forbidden Activities

- Copying someone’s work verbatim (classmate or otherwise)
- Copying someone’s work and obfuscating its source

Lab Topics

1. Linux
2. Version Control
3. C Programming
4. C Programming Project
5. Performance Measurement
6. Performance Optimization (compiler and programmer techniques)
7. Performance Optimization (Memory systems)
8. Network Programming 1 (Python)
9. Network Programming 2
10. Assembly Programming 1 (MIPS)
11. Assembly Programming 2
12. Assembly Programming 3

Lab 1 - Linux

Homework

Before the next class

1. **Skim "Virtual Machine Setup" tutorial instructions on website**
   - http://ecs-network.serv.pacific.edu/ecpe-170/tutorials/vm_setup
2. **Decide on what computer system you want to use for this class**
3. **Download all software**
   - Virtual machine installer (VMWare Player)
   - Linux .iso image (installer) – 64-bit version

Next Class - Linux Installfest

Tutorial Day

Objectives

- Follow the “Virtual Machine Setup” tutorial from website to install Linux
- Debug individual problems if needed
- Verify OS works
- **Email me screenshot as proof of success**

Next Class - Linux Installfest

I want you to be comfortable as professionals working independently to solve problems

If you complete the “Virtual Machine Setup” tutorial independently (**and email me a screenshot by Thursday morning**), you don’t need to attend Thursday’s class. Sleep in! *(Or come help out)*

I will still be here to answer all questions and solve problems

Next Class - Linux Installfest

Warning: Don’t skip class Thursday, and then tell me next Tuesday at Lab #1 that your OS doesn’t work!

Lab 1 - Linux

The first lab is next Tuesday

- Topic: Linux
- Crash course in command-line usage

Lab 1: Pre-Lab

- Show me the working command prompt in your Linux install. Hopefully you will have this done by end-of-class Thursday
- **Pre-Lab is due at the start of the lab**

Bring Laptop!

Every class – bring your laptop!

Questions?