Northern Arizona University • College of Engineering, Forestry, and Natural Sciences
School of Informatics, Computing, and Cyber Systems

Course Information

Catalog Description:	Binary representation of information in digital computers. An introduction to digital logic design, computer architectures, microprocessor architectures and assembly language programming.
Broad Topics:	Bitwise Operators, Digital Logic, Assembly Language, Computer Architecture, and C++.
Prerequisites:	CS 126
Co-requisites:	None
Skill Level:	Introductory
Credit Hours:	3
Class Number:	LEC 3139
Meeting Times:	MWF 11:30-12:20, Engineering Building 69, Room 101
Final Exam Time:	Wednesday, May 10, 2017, 10:00am - 12:00pm
Required Texts:	• Null & Lobur, "The Essentials of Computer Organization and Architecture", 4th Ed. Jones and Bartlett, 2014. ISBN **9781284045611**. • Britton, "MIPS Assembly Language Programming", 1st Ed. Prentice Hall, 2004. ISBN **013602212X**. **-OR-** The MIPS supplement for the Null & Lobur book, available at http://www.jblearning.com/catalog/Details.aspx?isbn13=9781284074482 (Select the Student Resources tab to order the supplement.)
Web Page:	BbLearn (bblearn.nau.edu)

Instructor Information

Instructor:	Patrick Kelley, BSCS
Office Hours:	Engineering Bldg. Rm 243 M, W, F 8:00 - 10:00 Tu,Th 8:00 - 12:00 Tu,Th 1:00 - 3:00
Email:	Patrick.Kelley (at) nau.edu or webmaster (at) flion.com
Phone:	699-7455 (email preferred)
NAU Address:	Box 15600 Flagstaff, AZ 86011

Course Description

In this class we will take a look under the hood at the low-level languages and hardware that power the high-level languages we're used to working with.

Course Objectives

By the end of the semester you should be able to:

- Relate the history and evolution of hardware.
- Translate between the various data representations used by computers.
- Use high-level languages to manipulate binary values.
- Express hardware functionality using boolean algebra.
- Utilize the fundamentals of hardware design.
- Write programs in assembly language.
- Identify the design characteristics of different machine languages.
- Recognize the issues and constraints involved in memory management and I/O.

Specific learning outcomes are detailed at the end of this syllabus.

Schedule

Week 1	Intro, Boolean algebra (Null chapter 3)
Week 2	Gates & circuits (Null chapter 3)
Week 3	C/C++ Programming (not in book)
Week 4	Bitwise ops (not in book)
Week 5	Bitwise ops, masks Test 1 *(February 17)*
Week 6	Data representation (Null chapter 1-2)
Week 7	Binary arithmetic (Null chapter 1-2)
Week 8	ISA's & memory (Null chapter 5 & 6)
Week 9	Assembly basics (Britton chapter 1 & 4) Test 2 *(March 10)*
Week 10	Assembly ops (Britton appendix A)
Week 11	Assembly ops
Week 12	Loops and jumps (Britton chapter 2) Test 3 *(April 14)*
Week 13	Functions (Britton chapter 5)
Week 14	Functions and stack (Britton chapter 6)
Week 15	I/O & system software (Null chapter 7 & 8)
Week 16	Final Exam *(5/10/2017 10:00 to 12:00)*

Coursework

The coursework includes the following assignments and tests:

- 10 Homework Assignments at 10 pts each (17% of total grade)

  Homework assignments involve research using book and online resources to answer specific questions. They help to fully prepare you for and familiarize you with the current lecture topics. Points are awarded for correct answers.

- 10 Programming Projects at 20 pts each (33% of total grade)

  Programming projects are where you put the knowledge you've gained into practice, transitioning from the theoretical to the practical with hands-on experience. Points are awarded based on the completeness and quality of your work and the thoroughness of your project report.

- 3 Tests: 60 pts each (30% of total grade)
  1 Final Exam: 120 pts (20% of total grade)

  The tests and the final are an incentive for you to ensure you fully understand the topics being covered - as well as demonstrating that fact to the instructor. Points are awarded for correct answers.

- Total: 600 pts (100%)

Your class grade is based on the standard scale of points earned: 90%=A, 80%=B, 70%=C, 60%=D, below 60%=F. **No grades are curved or dropped**, though there are opportunities for extra credit. Projects are individual effort.

Assignments are due according to the posted schedule in Bb Learn.

Extra Credit

There will be a number of extra credit possibilities that involve doing extra work on assignments. Besides that, you can also get +2 points on any project by turning it in at least two days early (ex. If it is due midnight Friday, then it must be in by midnight Wednesday to qualify for extra credit). **Note that your project must fully work to receive the extra credit**.

Late Policy

Project assignments are accepted up to a week late at a pro-rated 40% point penalty. Homework assignments are not accepted after the due date. The reason for this is that homework solutions are posted online after the due date.

If you miss a test or know you *will* miss an assignment or test, discuss the matter with me as soon as possible. Preferably beforehand so we can make other arrangements.

Attendance

Regular attendance is expected. Don't be late, and don't leave until class is dismissed. Roll isn't taken, but each lecture you miss seriously jeopardizes your overall comprehension of the material and your chances to do well.

Final Exam Policies

- If you score less than 50% on the final exam, your final class grade (which includes the final exam grade) will be reduced by one letter grade. This means that doing very poor on the final can severely hurt your class grade. If you had 480/480 points ("A") before the final and made 50/120 points on the final, your final class average would be 530/600 (88%) and your final grade would be "C".
- On the other hand if you do well in the class you get to take the Easy Final Exam, where all you have to do is sign your name on the day of the final. To qualify for the Easy Final you must meet ALL these conditions:
    - Have an "A" average at the end of the semester (90% or 432/480 points before the final).
    - Have scored 54/60 points or better on EACH of the first three tests.
    - Have completed all 10 projects with a grade of 16/20 or better.
    - Have turned in all 10 homework assignments with a grade of 8/10 or better.

Lectures and the Book

The lecture topics follow the same general outline as the books. However, the lecture complements the book rather than being a mirror of it. If you *only* read the book or *only* pay attention to the lecture you're likely to end up missing some key concepts. To get the most from the class, read each chapter before we discuss the corresponding topic in the lecture, then use the lecture as an opportunity to reconsider the key points of the material and ask questions on anything you're confused on.

Plagiarism and Cheating

Grades are a way to motivate students and to evaluate students' mastery of a subject and their ability to get work done. The grades you get are not themselves truly important, but instead are representative of your knowledge, capabilities, and work ethic, and ***those*** are the things that matter. If you plagiarize source code, fabricate results, make fraudulent claims, or attempt to cheat in any way, you are misrepresenting yourself, your level of understanding, your capabilities, and your ability to accomplish things. It is dishonest and unethical.

Anyone who plagiarizes, copies, fabricates, or cheats will at the *least* receive a zero on that assignment or test.

Consulting with others and using their advice on projects is fine. However, the programs you submit should be your own work that you thoroughly understand and are entirely responsible for. Don’t share code, just ideas.

Bb Learn

Most assignments and handouts will only be available on Bb Learn - they will not be handed out in class. In general, assignments will be posted by Monday and due the following week. Projects will be posted farther in advance but count on spending only a week on any project. Any clarifications, corrections, and announcements will be posted on Bb Learn.

I expect every student to follow the class progress in Bb Learn. Assignments and projects will be handed in via Bb Learn and your grades will be posted there. Read the assignment policies at http://www.cefns.nau.edu/~pek7/Common/assignment.pdf to see how I expect work to be submitted. Failure to follow the guidelines will affect your grade.

University Policies

There are a number of university policies that govern your education and safety that all students should be aware of. These are:

- Safe Working and Learning Environment
- Students With Disabilities
- Accommodation of Religious Observance And Practice
- Institutional Review Board (And Use Of Human Subjects)
- Academic Dishonesty
- Medical Insurance Coverage For Students
- Classroom Management
- Evacuation Policies

You will find a complete description of each policy here:

http://www.cefns.nau.edu/~pek7/Common/policies.pdf

EXTENDED COURSE DESCRIPTION

1. Explanation of Prerequisites
   CS 126
   - An ability to program alone and with a team.
   - An ability to solve small to medium sized problems with structured programming.
   - An ability to employ the software design process and software design tools to solve problems with computation.

2. Core Topics

- History of Computing Hardware
  - Organization vs. Architecture
  - Principle of Equivalence of Hardware and Software
  - Generations of Hardware
    - Analog (Gen 0)
    - Digital (Gen 1 – 5)
  - Computer Level Hierarchy
  - Von Neumann Model
  - Components of a computer system
- Boolean Algebra
  - Boolean functions and operators
  - Truth Tables
  - Boolean Identities and Simplification
  - Canonical Forms
  - Karnaugh Maps
- Digital Logic
  - Gates
  - Combinational Circuits
    - Adders
    - Multiplexers
    - Decoders
  - Sequential Circuits
    - Clocks
    - Flip-flops
    - Finite State Machines
    - Registers and Memory
    - Counters
- C/C++
  - Basic Language Syntax/Structure
  - Pointers and Memory
  - Bitwise Data Operators
  - Organization methods

- Data Representation
  - Unsigned data and Operations
  - Signed data and Operations
    - Sign-Magnitude
    - One's Complement
    - Two's Complement
  - Floating Point Representation
  - Character Sets
    - BCD/EBCDIC
    - ASCII
    - Unicode
- Instruction Set Architecture (ISA)
  - Opcodes and Operands
  - Instruction Formats
- Logical Memory Organization
  - Address Size
  - Code and Data Storage
  - Arrays and Strings
  - Endianness
- Assembly Language (MIPS)
  - Basic Language Syntax/Structure
  - HLL Construct Implementation
    - Conditional Statements
      - Status Register
      - CASE
      - Short-circuit evaluation
    - Loops (WHILE, DO WHILE, FOR)
  - Simple Procedures (register preserved parameters)
  - Stack-based Procedures
- System Architecture
  - Storage
    - Cache
    - Virtual Memory
    - RAID
  - I/O
    - Interrupts
    - Memory-mapped devices
  - System Software
    - Micro-kernel vs. Monolithic
    - Multi-tasking
    - Virtual Machines
    - Programming tools (Linking and Dynamic Linking)

3. Tools

   Tools used in this class include:

   - Logisim (a simple logic circuit editor/emulator)
   - A virtual Ubuntu development environment at CodeAnywhere.com
   - MARS or SPIM (free MIPS emulators)
   - Any text editor for creating assembly files

4. Learning Outcomes

L1. An ability to recognize the basic principles of computer organization and architecture.

By the end of this course students should be able to:

a. Describe the difference between organization and architecture.
b. Describe and apply the principle of equivalence of hardware and software.
c. Recognize the generations of computer hardware.
d. Recognize the levels of hierarchy in a modern computing system.
e. Understand von Neumann architecture and be able to trace the steps in processing.

This outcome supports BSCS ABET Student Outcomes A, E and K.
This outcome supports BSACS ABET Student Outcomes A, E and K.

L2. An ability to generate reports reflecting progress on assigned projects.

By the end of this course students should be able to:

a. Write and organize a report in a clear and engaging manner.
b. Describe in writing the project outcomes.

This outcome supports BSCS ABET Student Outcome G.
This outcome supports BSACS ABET Student Outcome G.

L3. An ability to work with Boolean algebra

By the end of this course students should be able to:

a. Generate truth tables for Boolean functions.
b. Simplify Boolean functions using the rules of Boolean algebra.
c. Simplify Boolean functions using Karnaugh Maps.
d. Expand Boolean functions into canonical forms.

This outcome supports BSCS ABET Student Outcomes A, E, and K.
This outcome supports BSACS ABET Student Outcomes A, E and K.

L4. An ability to create digital logic circuits

By the end of this course students should be able to:

a. Express a Boolean function in combinational digital logic.
b. Design multiplexers, decoders, and adders.
c. Express sequential circuits with a truth table and a logic diagram.

d. Express a sequential circuit as a finite state machine.
e. Use flip-flops with combinational digital logic to create counters, registers, and simple memory.

This outcome supports BSCS ABET Student Outcomes A, E, and K.
This outcome supports BSACS ABET Student Outcomes A, E, and K.

L5. An ability to use C/C++ for programming with an emphasis in bitwise operations.

By the end of this course students should be able to:
a. Write simple C/C++ programs.
b. Use appropriate data types and C-style type conversions.
c. Use C-style pointer reference and dereference syntax.
d. Use and interpret the results of the bitwise operators in C/C++.

This outcome supports BSCS ABET Student Outcomes A, E, and K.
This outcome supports BSACS ABET Student Outcomes A, E, and K.

L6. An ability to interpret and use binary data representation.

By the end of this course students should be able to:
a. Convert between binary, octal, decimal and hexadecimal.
b. Convert signed numbers between binary and decimal using sign-magnitude, 1s complement, and 2s complement.
c. Do binary addition, multiplication, subtraction, and division.
d. Convert rational numbers between bases.
e. Convert binary representation of rational numbers to decimal format.
f. Do addition, multiplication, subtraction, and division of rational numbers in binary representation.
g. Convert binary character data to BCD/EBCDIC, ASCII, and Unicode.

This outcome supports BSCS ABET Student Outcomes A, E, and K.
This outcome supports BSACS ABET Student Outcomes A, E, and K.

L7. An ability to create and interpret instruction set architectures

By the end of this course students should be able to:
a. Decipher machine language instruction formats.
b. Organize opcodes and instruction formats to create an ISA.

This outcome supports BSCS ABET Student Outcomes A, E, and K.
This outcome supports BSACS ABET Student Outcomes A, E, and K.

L8. An ability to understand and use different memory architectures.

By the end of this course students should be able to:
a. Relate memory space to address size.
b. Interpret data stored in memory, including arrays.

c. Interpret large data affected by Endianness.

This outcome supports BSCS ABET Student Outcomes A, E, and K.
This outcome supports BSACS ABET Student Outcomes A, E, and K.

L9. An ability to use assembly language

By the end of this course students should be able to:

a. Write and execute programs in assembly language.
b. Trace program execution and memory usage.
c. Write conditional statements for efficient processing.
d. Write compound conditionals and take advantage of short-circuit evaluation.
e. Identify and write assembly constructs equivalent to high-level language constructs, such as loops and case statements.
f. Write simple called procedures using register parameter passing.
g. Write advanced procedures using stack frames.

This outcome supports BSCS ABET Student Outcomes A, E, and K.
This outcome supports BSACS ABET Student Outcomes A, E, and K.

L10. An ability to work with advanced architectural features.

By the end of this course students should be able to:

a. Analyze hit/miss ratios and speedup ratios for cache memory.
b. Interpret addressing and page tables in virtual memory.
c. Describe various I/O strategies and how they are used.
d. Describe the steps in interrupt handling.
e. Identify the various RAID levels and describe their advantages.
f. Describe the differences between micro-kernel and monolithic systems and the advantages of each.
g. Describe various task-switching strategies for multi-tasking.
h. Describe virtual systems and their advantages.
i. Describe the steps of language compilation and linking.
j. Identify the generations of computer languages and their characteristics.

This outcome supports BSCS ABET Student Outcomes A, E, and K.
This outcome supports BSACS ABET Student Outcomes A, E, and K.