Computer Science 250
Computer Organization and Assembly Language Programming
Spring 2016

Professor: James Wilkinson
Office: 306 Harbor Walk East
Phone: (95)3-8160
email: the.doc@twc.com (I keep up with this one)
wilkinsonj@cofc.edu (I sometimes look at this one)

Office hours: TTH 1:30 – 2:00, 5:00 – 5:30
and by appointment

Class Meeting Times:
Section 2 – 2:10 - 3:25 TTH in Harbor Walk East 300
Section 1– 3:35 - 4:50 TTH in Harbor Walk East 300

Class web site: http://wilkinsonj.people.cofc.edu
MARS download site: http://courses.missouristate.edu/KenVollmar/MARS/index.htm
Slide-show link: https://drive.google.com/?authuser=0#folders/0B_Gsp_3VEiUQbTZaRjNKQXM3b00
Berkeley lectures link: https://www.youtube.com/playlist?list=PL-XXv-cvA_iCl2-D-FS5mk0jFF6cYSJs_

Text: Patterson and Hennessy- Computer Organization and Design 5th ed.
(Morgan Kaufman ISBN-13: 978-0-12-407726-3)

Prerequisites are CSCI 220 and 220L Co-requisite is Math 207

Course Description:

An introduction to an assembly language and its implementation in hardware. Topics include the binary and hexadecimal numeration systems, the fetch-execute cycle, the components of the central processing unit, floating point processing, memory, compiling, assembling, linking, cache memory, virtual memory, pipelining. Programming exercises are developed in the assembly language of a commonly available processor.

Course Policies:

1. Attendance: I strongly encourage you to attend all classes. Regardless of actual attendance, you are responsible for announcements made in class, assignment due dates, etc. There will be two tests and a comprehensive final exam, attendance at all of which is mandatory. I will announce test dates at least a week in advance.

2. Disability Accommodation: If you feel the need for an accommodation due to a disability, then speak to me individually to discuss your specific needs. For additional help please contact the Center for Disability Services at "http://www.cofc.edu/~cds/".

3. Quizzes: We will begin each class with an eight-minute quiz. Since class time is valuable, I will have to enforce the time limit strictly. You will have to do your own arithmetic; calculators are not permitted.

4. Programs: There will be programming assignments in the MIPS assembly language. These will be used to reinforce the concepts in the course; we're not trying to turn you into assembly-language programmers. Each programming assignment will have a due date and time, and no assignments will be accepted late. These assignments will be administered and graded by computer. Some test questions will be based on these programs.

5. Test Average: Tests 1 and 2 – 25% each; Final exam – 50%.

6. Final Grade Computation: The final grade will be computed as follows: test average – 80%, programs – 10%, quizzes – 10%.

7. Grading Scale: A: 90 – 100; B: 80 – 89; C: 70 – 79; D: 60 – 69; F: <60. Plusses and minuses may be given at the instructor's discretion.

Honor Code and Academic Integrity: Lying, cheating, attempted cheating, and plagiarism are violations of our Honor Code that, when identified, are investigated. Each instance is examined to determine the degree of deception involved. Incidents where the professor believes the student's actions are clearly related more to ignorance, miscommunication, or uncertainty, can be addressed by consultation with the student. We will craft a written resolution designed to help prevent the student from repeating the error in the future. The resolution, submitted by form and signed by both the professor and the student, is forwarded to the Dean of Students and remains on file.
Cases of suspected academic dishonesty will be reported directly to the Dean of Students. A student found responsible for academic dishonesty will receive a XF in the course, indicating failure of the course due to academic dishonesty. This grade will appear on the student's transcript for two years after which the student may petition for the X to be expunged. The student may also be placed on disciplinary probation, suspended (temporary removal) or expelled (permanent removal) from the College by the Honor Board.

It is important for students to remember that unauthorized collaboration--working together without permission-- is a form of cheating. Unless a professor specifies that students can work together on an assignment or test, no collaboration is permitted. Other forms of cheating include possessing or using an unauthorized study aid (such as a PDA), copying from another's exam, fabricating data, and giving unauthorized assistance.

Students can find a complete version of the Honor Code and all related processes in the Student Handbook at
http://www.cofc.edu/studentaffairs/general_info/studenthandbook.html.

Course Goals: The intention of this course is that students will attain the following knowledge and skills:

1. Be familiar with the von Neumann model of a computer system, and understand the basic operation and terminology associated with various components (the CPU, memory organization, disk drives, and various IO devices).

2. Be familiar with the basic architecture of modern processors, and to make objective comparisons of different types of architectures such as RISC and CISC based on performance measures (for example cpu cycles per instruction and cycle times).

3. Be able to express numbers in the decimal, binary, and hexadecimal number systems and convert numbers between those systems. The student should also be able to describe the implementation of two's complement number representation on typical machines.

4. Understand typical methods used to implement standard data types (integer data, logical data, character and string data, floating-point data) at the machine level.

5. Demonstrate methods of accessing information in machine memory using direct or indirect addressing schemes, and describe various memory management schemes such as virtual memory.

6. Understand basic IO techniques including memory-mapped IO vs special IO instructions, and interrupt-driven IO vs polling.

7. Write correct assembly-language programs (in MIPS assembly language) for simple tasks.

8. Understand the mapping from basic high-level source code features such as arithmetic expressions, array references, while loops, and if statements to assembly language.

9. Understand a subroutine calling convention and the role of a stack frame in this convention. Implement such a convention in a two-level calling sequence (that is, main calls sub1 calls sub2).

10. Understand the compilation, assembly, and linking processes.