

MC160-Computer Organization and Assembly Language

FALL, 2003

This syllabus is available online at www.cs.bc.edu/~cbrown

Instructor
Teaching Assistant
Class Meetings
Course Content
Textbook and Software
Required Background
Handouts and Lecture Notes
Required Work
Grades
Getting Help
Academic Honesty

Instructor

Craig Brown
Fulton 410/414
E-mail: brownqy@bc.edu
Office Hours: M,W 3-4; Tu 11-1; Phone: 617-552-6371

Teaching Assistant

Anya Ioffe
e-mail: ioffe@bc.edu
Office Hours: Tu. & Thur. 10:30AM-12:30PM, Fulton 160

Class Meetings

The class meets Monday, Wednesday, and Friday from 2:00PM-2:50PM in Fulton 415

Course Content

To the user of an application program, for example a text-processing package, the computer appears to be an electronic text-processing machine, as though it were designed to interpret and execute commands to place a character in a document, change fonts, print, etc. To someone writing such application software in a high-level language like C, the computer appears to be a machine for interpreting and executing C programs. However, something else is happening at a still lower level: The C program is translated into a sequence of machine code instructions which are loaded into the computer's memory and executed. From this point of view, the computer is a machine for interpreting and executing machine code instructions.

There is a lower level than this: Machine code instructions are interpreted and executed by electrical circuits that perform very rudimentary logical operations. From this point of view, a computer is a vast network of switches.

This course is a look under the hood of the computer. It is a remarkable fact that despite the enormous changes in the technology of computer manufacture, the fundamental logical organization of most of the

computers in use today is the same as it was fifty years ago: A large bank of memory communicating with a processor that executes, one at a time, instructions stored in the memory. We will begin the course with a look at a simple version of this fundamental computer model (the "von Neumann machine"). Since it is critically important to understand how numerical data are represented in computer memory, we will spend some time studying number representation systems, both at the start of the course when we look at integer representations, and near the end when we look at the floating point representations.

Most of the first half will be devoted to programming in assembly language, which is a symbolic code for writing machine instructions. Different processors have different machine codes, and consequently any course in assembly language is bound to be highly machine-specific. In this course we will use the set of machine instructions for the MIPS assembler which supports MIPS R2000/R3000 processors. In spite of the differences between machines, the machine codes of different processors do resemble one another to a large extent, so what you learn in this course can easily be applied if you ever have to write in the machine or assembly language of another processor.

Much of the remainder of the course (approximately the second half of the semester) will be devoted to the study of the digital logic level of computer organization. We will see how basic logic gates are combined to build arithmetic units, registers and memories, and ultimately, computers. Using this "drill down" approach should provide a meaningful learning experience for anyone starting with a knowledge of higher level programming languages.

Here is a brief (approximate) outline of the course topics:

- Introduction to the von Neumann machine
- Positional Number Systems
- The MIPS Processors and Basic Assembly Instructions
- Addressing Modes and Instruction Formats
- Branching and Looping Instructions
- Subroutine Linkage and Recursion
- Exception Processing
- Floating Point Representation and Coprocessor Instructions
- Digital Circuits

Required Textbooks and Software

Tanenbaum, Andrew S. 1999. *Structured Computer Organization*. Upper Saddle River, New Jersey: Prentice Hall. ISBN: 0-13-095990-1. Available through the BC bookstore.

Britton, Robert. 2003. *MIPS Assembly Language Programming*. Prentice Hall. ISBN:0-13-142044-5. Available through the BC Bookstore.

Professor Howard Straubing from Boston College's Computer Science Department has prepared a series of lecture notes covering all parts of the course and I will depend heavily on his notes as a reference. You can gain access to these notes via the MC160 WebCT site.

During the first week of the course you will work with a simulated computer that lives on an Excel spreadsheet. For the digital logic portion of the course, we will use a very nice freeware program, called Digital Works, for simulating digital circuits. This Freeware can be downloaded from http://www.cs.bc.edu/~cbrown/mc160f03/misc/dw20_95.exe. For the assembly language portion, you will also need an environment in which you can edit, assemble, link and run assembly-language programs. We will use a Freeware MIPS simulator called SPIM to support the required assembly language programming. This software is available for download from <http://www.mkp.com/cod2e.htm>. This site is also an excellent reference for computer architecture and assembly language programming information.

Optional Textbooks

Patterson, David A. and John L. Hennessy 1997. *Computer Organization and Design – The Hardware/Software Interface*. Morgan Kaufmann Publishers, Inc. ISBN: 1-55860-428-6. Available through the Internet. This book is particularly valuable for its Appendix A on the MIPS assembler and SPIM simulator. The Appendix A information is also available at <http://www.cs.wisc.edu/~larus/SPIM/cod-appa.pdf>. The book should also be on reserve in the BC Library.

Sweetman, Dominic. *See MIPS Run*. Morgan Kaufmann Publishers, Inc., San Francisco, 1999. ISBN: 1-55860-410-3. This book should also be on reserve in the BC Library.

Required Background

I expect that all students have taken MC140 (Computer Science 1) or an equivalent course. I strongly recommend that all students have taken MC141 (Computer Science 2) as well, or are taking CS2 concurrently. This will become particularly important when we study subroutine linkage, recursion, and indirect addressing in assembly language.

Required Work

Homework

There will be about nine weekly homework assignments. These will consist of programming problems and written problems. I will post solutions to each assignment a few days after the due date. Late homework will not receive any credit without an official excuse from the Dean. WebCT will also prevent multiple submissions of the same homework assignment so MAKE SURE WHAT YOU SUBMIT IS CORRECT BEFORE YOUR SUBMISSION!

I encourage you to discuss the homework problems both among yourselves and with me. However, the work you hand in must be your own. This means that it is all right for one student to give another part of the essential idea for solving a problem in the course of these discussions; it is NOT all right for one student to give another one hundred lines of code to copy and hand in as part of an assignment. All the assignments and solutions will be distributed through this Web site, by clicking on the appropriate link below.

Weekly homework assignments and Instructions for electronic submission of assignments will be posted on the MC160 WebCT site. Unless otherwise instructed “ALL” homework must be submitted via WebCT utilizing using computer based generation tools such as Microsoft Word, Excel, etc.

Exams

There will be two in-class midterms (exact dates to be determined) and a final exam at the regularly scheduled hour for courses in this time slot. I will probably throw in about three or four quizzes as well. Exams and quizzes will generally be open notes and open books and portions of the exams will test your ability to program using MIPS assembly language as well as design basic digital circuits.

Handouts and Lecture Notes

Refer to WebCT for access to handouts and notes. <http://webct.bc.edu:8900>

Grades

To be eligible for a passing grade in the course you must satisfactorily complete a minimum of 60% of the weekly homework assignments, handed in before the solutions are posted. Your course grade will then be computed from the scores on homework (30%), the in-class midterms (30%) and quizzes (5%), and the

final exam (35%). Course grades may be subject to normalization. More information on grading criteria may be found via WebCT under the "LETTER GRADE CRITERIA" section.

Getting Help:

If you need assistance, there are several places you can get help on campus. My office location and hours are at the beginning of this syllabus, and I can be available at other times as well. My office hours and the TA's office hours are published in this syllabus.

If you feel you need tutoring, please contact the Academic Development Center in the O'Neill Library. They tend to be saturated near the end of the semester, so contact them early if you'd like help.

Academic Honesty

I expect you to abide by the standards of academic honesty set in the student guide. Cheating and plagiarism are not worthy of Boston College students. You may discuss your homework with your peers, but your submitted solutions must involve only your individual effort.

In addition, I expect that you are familiar with the computer ethics policy authored by the Office of the Dean for Student Development, which is also part of the student guide. If you don't have a student guide, both standards (academic honesty and computer ethics) are available on-line through: http://www.bc.edu/bc_org/avp/enmgt/stserv/acd/univ.html#integrity, Section VI. Policies and Procedures (items related to Community Standards, Academic Honesty, Computer Ethics Policies and Integrity and Protection of Technological and Information Resources). Please familiarize yourself with them.