Course Description

Instructor: Chintan Patel
Office: ITE 322
Office Hours: Mon & Wed 10:00am–11:30am
Telephone: 455-3963
Email: cpatel2@csee.umbc.edu

The TAs' office hours will be announced at a later date.

Time and Place. Monday & Wednesday 5:30pm – 6:45pm, ITE 233.

Textbooks.

- *Principles of Computer Architecture,* Murdocca and Heuring, Prentice-Hall, 2000.
- *Linux Assembly Language Programming,* Neveln, Prentice-Hall, 2000.

Course Web Page: <http://www.cs.umbc.edu/~cpatel2/>

Prerequisites. You should have mastered the material covered in the following courses: CMSC 202 Computer Science II and CMSC 203 Discrete Structures. In particular, we will assume that you have had extensive programming experience in C/C++. Also, you must be familiar with and be able to work with truth tables, Boolean algebra and modular arithmetic.

Objectives. The purpose of this course is to introduce computer science majors to computing systems below that of a high-level programming language. The material covered can be broadly separated into the categories of assembly language programming and computer organization. Under the heading of assembly language programming students will be introduced to the i386 instruction set, low-level programming, the Linux memory model, as well as the internal workings of compilers, assemblers and linkers. Topics under computer organization include digital logic design (combinational circuits, sequential circuits, finite state machines) and basic computer architecture (system bus, memory hierarchy and input/output devices). A secondary goal of this course is to prepare computer science majors for CMSC 411 Computer Architecture.

Grading. Your final grade will be based upon 5 homework assignments (20% total), 3 short programming assignments (12% total), 1 long programming assignment (8%), 1 in-class lab assignment (4%), 2 circuit simulation exercises (8% total), a midterm exam (24%) and a final exam (24%). Your grade is given for work done *during* the semester; incomplete grades will only be given for medical illness or other such dire circumstances.

Lecture Policy. You are expected to attend all lectures. You are responsible for all material covered in the lecture as well as those in the assigned reading. However, this subject cannot be learned simply by listening to the lectures and reading the book. In order to master the material, you need to spend time outside the classroom on the programming assignments, simulation exercises and homework assignments.

Due Dates. There will be homework or exercises due on most weeks. Homeworks are due at the beginning of lecture. Exercises and projects turned in via online submissions are due 1 minute past 11:59pm of the due date. ***With one exception, late homework, exercises and programming assignments will not be accepted — this is to allow for timely grading and discussion of the solutions. The exception is that each student may submit one assignment (of any kind) up to one week late during the semester.***

Academic Integrity. You are allowed to discuss the homework assignments with other students. However, exercises and projects must be completed by individual effort. Furthermore, you must write up your homework ***independently***. This means you should only have the textbooks and your own notes in front of you when you write up your homework — not your friend's notes, your friend's homework or other reference material. You should not have a copy of someone else's homework or project ***under any circumstance***. For example, you should not let someone turn in your homework. ***Cases of academic dishonesty will be dealt with severely.***

Exams. The exams will be closed-book and closed-notes. The date for the midterm exam is Thursday, October 16. The final exam will cover the material from the second part of the course. The date and time of the final exam is Tuesday, December 16, 10:30am to 12:30pm.

Advising Note. This course is a replacement for CMSC 211 Assembly Language Programming and CMSC 311 Computer Organization which are no longer offered at UMBC. However, computer science majors who take this class must also take CMSC 411 Computer Architecture to satisfy the requirements of a BS degree in computer science. CMSC 313 by itself will not be sufficient for graduation — even if you've taken CMSC 211 or CMSC 311 previously.

Course Syllabus

We will follow two textbooks: *Principles of Computer Architecture*, by Murdocca and Heuring, and *Linux Assembly Language Programming*, by Neveln. The following schedule outlines the material to be covered during the semester and specifies the corresponding sections in each textbook.

Date	Topic	M&H	Neveln	Assign	Due
We 08/27	Introduction & Overview	1.1-1.8	1.1-1.6		
Mo 09/01	*Labor Day*				
We 09/03	Data Representation I	2.1-2.2, 3.1-3.3	2.4-2.7, 3.6-3.8	hw1	
Mo 09/08	Data Representation II				
We 09/10	i386 Assembly Language I		3.10-3.13, 4.1-4.8	hw2, proj1	hw1
Mo 09/15	i386 Assembly Language II		6.1-6.5		
We 09/17	i386 Assembly Language III			proj2	hw2, proj1
Mo 09/22	i386 Assembly Language IV				
We 09/24	Examples			proj3	proj2
Mo 09/29	Machine Language		5.1-5.7		
We 10/01	Compiling, Assembling & Linking	5.1-5.3			
Mo 10/06	Subroutines		7.1-7.4		
We 10/08	The Stack & C Functions			proj4	proj3
Mo 10/13	Linux Memory Model	7.7	8.1-8.8		
We 10/15	Interrupts & System Calls		9.1-9.8		proj4
Mo 10/20	**Midterm Exam**				
We 10/22	Introduction to Digital Logic	A.1-A.2	3.1-3.3		
Mo 10/27	Transistors & Logic Gates	A.3-A.4		hw3	
We 10/29	In-class Lab I				
Mo 11/03	Boolean Functions & Truth Tables	A.5-A.9		hw4	hw3
We 11/05	Circuits for Addition	3.5			
Mo 11/10	Circuit Simplification I	B.1-B.2		hw5	hw4
We 11/12	Combinational Logic Components	A.10			
Mo 11/17	Flip Flops	A.11		digsim1	hw5
We 11/19	In-class Lab II				
Mo 11/24	Finite State Machines	A.12-A.15			
We 11/26	Circuit Simplification II	B.3			
Mo 12/01	Finite State Machine Design			digsim2	digsim1
We 12/03	More Finite State Machine Design				
Mo 12/08	I/O & Memory	7.1-7.6, 8.1-8.3			digsim2
Mo 12/15	**Final Exam 6pm-8pm**				

Policy on Programming Projects and Exercises

Critical programming skills cannot be learned by attending lecture. You should budget enough time to work on the programming assignments as well. Please consult the time table given on the syllabus and plan ahead. Programs are due by midnight (1 minute after 11:59pm) of the due date. Programs will be submitted using the `submit` system running on the GL machines. Late assignments will not be accepted (with the one exception noted in the course description). Programs will be graded on five criteria: correctness, design, style, documentation and efficiency. So, turning in a project that merely "works" is not sufficient to receive full credit.

For this course, programming projects must be developed using the NASM assembler for the Linux operating system running on an Intel Pentium CPU. This arrangement is not compatible with other flavors of UNIX, with Linux running on non-Intel CPUs or with assemblers for Windows 95/98/2k/ME/XP/NT. When in doubt the UMBC machine `linux.gl.umbc.edu` will be the final arbiter of what constitutes a working program. You may work on your own machines running Linux, but you will have to be your own system administrator. None of the instructors, TA or support staff at OIT will be available to help you install or debug Linux.

Cheating

Read this section carefully! It describes what constitutes cheating for this course. If you have questions, ask the instructor. Ignorance will not be accepted as an excuse after the fact.

All programming assignments and circuit simulation exercises must be completed by your own individual effort. You should never have a copy of someone else's program either on paper or electronically under any circumstance. Also, you should never give a copy of your program or circuit, either on paper or electronically, to another student. This also means that you cannot work on the programming assignments or circuit simulation exercises together. Cases of academic dishonesty will be dealt with severely. Egregious cases of cheating will be reported as a major infraction. In this case, you will not be allowed to drop the course. Also, a major infraction would appear as a permanent part of your student record and would be seen by potential employers when they ask for an official copy of your transcript.

We will be using special software to check for cheating. The software is quite sophisticated, has been tuned for assembly language programs and has surprised some students in the past. We will, of course, not release the details of the internal workings of this cheat-checking software, but you are forewarned that there is no difficulty in comparing every pair of submitted projects.

Finally, you are also warned that if your program is turned in by someone else, then, at a minimum, both you and the person who copied your program will receive a 0 for that assignment. This includes substantially similar programs. Furthermore, all parties concerned will have their prior programs checked for cheating. So, if you cheat on the last assignment, you can lose all the points from all of your assignments — even if you did all the work and just let other people copy from you.