Course Information and Standards
Computer Architecture
COMP 3050

University of Massachusetts Lowell
Department of Computer Science
Fall Semester 2023

Time and location:	T R 9:30 Sect. 201 and 11:00 Sect. 202, Falmouth 313
Instructor:	Prof. W. Moloney
Office:	Dandeneau 347
Office hours:	T R 7:30 AM – 9:15 and by appointment
Email:	bill@cs.uml.edu
Class Web site:	www.cs.uml.edu/~bill/cs305

1. Course Description:

An examination of the basic functional components of a computer system including the CPU, memory systems, and I/O systems. Each of these three areas will be developed in detail with a focus on the system design and component integration. Topics will include CPU control and ALU operation, computer timing, data address and I/O bus activity, addressing model, programmed and DMA I/O, and instruction sets and microcode.

All course materials used throughout the term will be posted to the class webpage:

www.cs.uml.edu/~bill/cs305

If you need to contact me out of class, please use my email address:

bill@cs.uml.edu

The programming assignments in this class must be completed on the mercury,cs,uml machines and submitted electronically according to the assignment submit document at:

www.cs.uml.edu/~bill/cs305/Assignment_Submit_Details.pdf

Programming assignments will be built and tested on ONLY the mercury systems, so you should do all of your development on mercury.cs.uml.edu. You can access mercury from anywhere on the internet using an ssh application, but you must first establish a VPN connection to UML before you attempt to ssh to mercury. See the IT material on connecting a VPN at:

https://www.uml.edu/it/services/get-connected/remote-access/

2. Prerequisites:

COMP 2010 Computing III, COMP 2030 Assembly, EECE 2650 Logic Design

3. Required Text:

Computer Systems: A Programmer's Perspective (3rd Edition), by R. Bryant and D. O'Hallaron, Addison Wesley, Pearson, 2016, ISBN 978-0-13-409266-9

A photo copied version of the text can be found at:
http://54.186.36.238/Computer%20Systems%20-%20A%20Programmer%27s%20Perspective%203rd%20ed.%20%282015%29.pdf

and a clean PDF version can be found at:
https://www.cs.sfu.ca/~ashriram/Courses/CS295/assets/books/CSAPP_2016.pdf

4. Grading:

Final grades will be based as follows:

Type	Number	Weight
Hour Exams	2	40 %
Final Exam	1	20 %
Programming Assignments	approx. 7	40 %

The programming assignments are to be coded using the `C' or `C++' programming language, or the programming tools of the MIC-1 system as described in class.

5. Lateness:

Assignments are graded from 0 – 10 points. For each one of our **class meeting days** that an assignment or project is late, 10 % of the total points (1 point) will be deducted from the points received. This will continue until midnight of the **fifth class meeting day after the assignment is due** (approximately 2 weeks, **except for the final two assignments** which must be submitted by **our last class meeting** day of the fall semester, **Thursday, December 14)**, after which the assignment or project **will no longer be accepted.** The cutoff date for each assignment is included in the posted class calendar (www.cs.uml.edu/~bill/cs305/Class_Calendar.pdf). **An assignment will be graded as zero points if not submitted by a cutoff date.**

6. Academic Dishonesty:

In this course, **all** work is to be **each student's own**. Students should therefore be familiar with the University's rules on academic dishonesty, which can be found in the *Bulletin of Undergraduate Studies* and in the *Schedule of Classes*. In particular, **plagiarism** will not be tolerated! Any student caught plagiarizing another's work will automatically receive a grade of **F** for the course. If you are unsure as to what constitutes plagiarism, it is your responsibility to check with the instructor. Other forms of dishonesty will result in similar actions. You may collaborate with your classmates on the design and results of the programs you will write in this course, but **each student must implement these programs alone**. Submission of **shared student code is not permissible**, and will result in a grade of **F** for the course. Help files are typically provided for each programming assignment, and students are encouraged to cut and paste useful code from these help files into their assignment submissions, but **all other code must be the specific work of each student**.

7. No Posting of Solution Code Policy:

You are **not** allowed to post solution code to problem sets assigned in this class in public places (e.g. Github). This includes your own solutions as well as solutions that may be provided by the instructors.

The University policy on academic integrity states that assisting students in their own acts of academic dishonesty is itself a violation of academic integrity. See Academic Misconduct Subject to Disciplinary Action, 1(f).

Please note that this is typical policy at premier computer science departments. E.g.:

- Princeton COS 126. *"Your work must never be shown or communicated to anyone who is taking COS 126 now or who might take COS 126 in the future. ... You must never place your work in any public location (including websites, leaving printouts in a classroom, etc.). ... The rules ... continue to apply even after this semester is over."*
- Harvard CS50. *"Not reasonable: Providing or making available solutions to problem sets to individuals who might take this course in the future."*
- MIT 6.01. ***"Students should never share their solutions (or staff solutions)** with other students, including through public code repositories such as Github."* (emphasis in the original)

Thus: **Do not publish your solutions to problem sets.**

Doing so will be considered an act of academic dishonesty and you will receive a grade of F for the course.

Note: You may save your work to code repositories provided that they are private and cannot be retrieved by others. I encourage you to sign up for Github's Student Developer Pack, which allows you to create private repositories (among other benefits). It's free to students.

8. Topical Outline:

The topics covered in this course include the following:

- Internal representations
- Machine level representation of programs
- Processor architecture
- Memory hierarchy
- Cache architectures
- Linking
- Exceptional control flow
- Hardware and software exceptions
- Operating System organization
- Processes and threads
- System call programming