Course Syllabus
ICS 51 - Introductory Computer Organization

Prerequisites: ICS 22 and Math 6B (or equivalents)


COURSE OVERVIEW
This course is a first introduction to computer system organization and assembly language programming. The computer system is presented as a hierarchy of levels or layers. Each layer is built on top of the previous one and abstracts away the details of the lower layer while providing additional capabilities to the layers above. This relationship aids in understanding the system and its programming by breaking it into more easily understandable sub-systems. It is also used in the design of computer systems to break it down into manageable tasks.

In our study of computer systems, the following levels are considered:

<table>
<thead>
<tr>
<th>Level</th>
<th>Studied in:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly Language</td>
<td>Labs</td>
</tr>
<tr>
<td>Operating System</td>
<td></td>
</tr>
<tr>
<td>Instruction Set Architecture</td>
<td>Main lecture course</td>
</tr>
<tr>
<td>Micro-architecture</td>
<td></td>
</tr>
<tr>
<td>Digital Logic</td>
<td></td>
</tr>
</tbody>
</table>

Above the assembly language level—the highest level in the hierarchy—are programs and data structures that implement applications. These are normally written in a high-level programming language, such as Java or C, and are translated into the lower levels by a compiler. Below the lowest level—the digital logic level—are transistors and other hardware elements that implement this level.

As indicated in the diagram, the main lecture course will focus on the four lower levels, which will be present in a bottom-up approach, staring the digital logic and ending with operating system concepts. The assembly language level will be handled separately and concurrently in the labs.
COURSE STRUCTURE

Lectures:
Each lecture is provided as a video recording using a PowerPoint presentation. The length varies based on the topic, ranging from 10 to about 30 minutes. Copies of the PowerPoint slides are available as notes on course web page. The lectures are based on selected sections of the book.

Homework assignments:
Each week’s lectures are accompanied by a set of homework assignments to practice the presented material. All homework assignments need to be turned in by the due date. The homework will not be graded for correctness (only for effort) but solutions will be posted after the due date.

Please note: The points earned for homework make up only a relatively small percentage of the total grade and are intended only as an incentive for you to complete and turn in the work on time. However, the problems on the weekly quizzes and the final exam will be very similar to those on the homework. Thus doing the homework is essential for doing well on the quizzes/exam!

Exams:
There will be a written exam at end of each 2-week period. It will cover the material of the past two weeks. There is no midterm or any comprehensive final exam.

Programming Labs:
The labs focus on the fifth level: assembly language programming. They run concurrently with the lecture course and their objective is to give you practical hand-on experience with computer organization. There are several lab-oriented lectures designed to teach you the principles of assembly language programming using a small subset of Intel 8088 instructions. This is followed by detailed explanations of the specific assignments to be carried out.

GRADING
Your grade in ICS 51 will depend on your performance on exams, homework, and lab assignments. The following weights will be used:

- Homework - 10%
- Lab - 30%
- Exams - 60%

The exam with the lowest grade will automatically be dropped.
The grades will be based on a curve. The approximate grade distribution will be:

- 25% A+-
- 25% B+-
- 30% C+-
- 15% D+-

Late work: A penalty of 10% per day will be imposed on all late work (zero credit after solutions have been posted)
MAIN TOPICS COVERED

Lecture course
- Hierarchical view of computer systems
- Number systems
- Gates and Boolean algebra
- Digital logic circuits
- Memory
- The data path
- Microinstruction control
- Implementation of a machine instruction set
- Pipelined architecture
- Caches
- Data and instruction types
- Expanding opcodes
- Error-detecting/correcting codes
- Virtual memory

Labs
- Assembly language programming