Catalog Description
This course introduces principles of computer organization: levels of computer organization, digital logic, microprocessing, machine language, assembly language, operating system processes, memory, interrupts, addressing, controls, paging, tasking, and linkage.

Prerequisite: MAT 202 Computer Science II.

Evan’s Comments
Computer Organization is a dynamic topic, but to understand where we are now it is essential to start at (very nearly) the beginning. In this course we will investigate a sort of “behind the scenes” look at a computer and how it operates. As we investigate the topics listed in the Catalog Description (above) we will consider questions like: Where have we been? Where are we now? Where might we go next? We will delve into the nuts and bolts (theoretical and literal) of the microelectronics of a computer. We will follow a piece of Java code all the way down to what happens with every tick of the system clock (and what happens between the ticks and the tocks). Throughout the course, we will also discuss contemporary advancements and issues.

Course Goals
The goals of this course are to enable each student:
- To recognize different levels of computer architecture
- To distinguish among the different types of memory
- To read and design digital logic systems and
- To construct (virtual) microelectronic machines
- To follow a piece of compiled Java code through the entire process of implementation

Objectives
In the process of taking this course each student will encounter:
- The complete computer model from the user interface to the microelectronics level
- The interaction of computer hardware and software (and the grey area in between)
- How information (numbers, instructions, data) is stored and accessed
- The microelectronics of digital logic and microcircuits
- How compiled Java code is implemented on the micro-architecture level
- Contemporary advancements and issues in computer organization
Course Information – MAT 304: Computer Organization

Required Materials:
Text: *Structured Computer Organization (5th Ed.)* by Tanenbaum (bring to class)
Technology: Internet and e-mail

Course Structure

- Course Overview
  
  I will assume that you have read the required material before class.

  We will loosely adhere to the text through Chapter 5 including a study of Appendices A and B. Along the way we will consider other sources to complement the material in the text – finding and presenting these sources will be our mutual responsibility. Most of this material is history and information more than problem solving techniques and specific skill building, so only selected topics will be studied in more detail in class.

- Class

  Attendance and class participation are required, expected and *essential*. Our class time will be spent mostly with lecture/discussion with a significant amount of the material presented by *you*. All I ask is that you treat everyone in the class with respect, and at times remember that I am the one in charge. It is *your* responsibility to be properly prepared for class.

- Assessment

  The most important aspect of this course will be active, mutual participation. There will also be several small assignments (to be turned in), and a final paper and presentation.

  Your final grade will be calculated by 50% participation, 25% small assignments, 25% final paper and presentation. I will schedule regular meetings to keep you apprised of your progress.

**Important Dates/Class Calendar**

<table>
<thead>
<tr>
<th>Month</th>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>January</td>
<td>18 (M)</td>
<td>Triad Day – MLK Day Celebration</td>
</tr>
<tr>
<td>March</td>
<td>13 – 21</td>
<td>Spring Break</td>
</tr>
<tr>
<td>March</td>
<td>26 (F)</td>
<td>Deadline to Withdraw (non-punitive)</td>
</tr>
<tr>
<td>April</td>
<td>7 (W)</td>
<td>Work Day (No Classes)</td>
</tr>
<tr>
<td>May</td>
<td>11 (T), 13(Th)</td>
<td>Student Presentations</td>
</tr>
<tr>
<td>May</td>
<td>13 (Th)</td>
<td>Last Day of Classes</td>
</tr>
</tbody>
</table>

An x86 processor is screening alone at billions of cycles per second to run the xau kernel, which is painstakingly working through all the posix-specified abstraction to create the draw system underlying cs x, which in turn is spinning itself to run firefox and its gsd renderer, which creates a flash object which renders dozens of video frames every second.

Because I wanted to see a cat jump into a box and fall over.

I am a god.