Syllabus – Spring 2008
CS 271 Computer Architecture
CRN 21221: MW 6:00PM - 7:15, ET 109

Instructor

Mark Temte, Ph.D.
Office: ET-125F
Office hours: MW 4:30 PM – 5:45, TR 1:30 PM – 2:45
Phone: 260-481-6181
Email: temte@ipfw.edu
Course web site: users.ipfw.edu/temte/

Course Description

P: CS 161 and MA 175. Introduction to computer organization and architecture. Fundamentals of digital logic, and representation of numeric and nonnumeric data. Assembly level organization and programming, including instruction formats, addressing modes, and subprogram call/return. Design of main memory and cache memory. Interrupt basics, interrupt-driven I/O, DMA, and bus protocoals. Processor organization, data paths, the control unit, microprogramming, pipelining, and performance enhancements. Multiprocessor and alternative architectures.

Course Learning Outcomes

(These support the ABET Program Outcomes found at the end of the syllabus, as indicated by the letters in parentheses.)

Upon successful completion of the course requirements, a student should be able to:

1. Demonstrate facility with radix number systems, two's complement arithmetic, and floating-point number representation (a, i).
2. Show familiarity with the Intel 8088 processor and memory model (b, i, j).
3. Show ability to translate basic high-level language control structures into equivalent assembly language statements and to use pseudoinstructions (c, i).
4. Perform assembly language I/O with text, numbers, and files in programs (c, i).
5. Decompose a software problem into a main program and subroutines and implement the solution in assembly language using proper subroutine linkage (c, k).
6. Demonstrate the ability to use the assembly language instruction set and appropriate addressing modes in a program involving a data table (c, i).
7. Show understanding of digital logic concepts and selected combinational logic and sequential logic circuits (a, i).

8. Show understanding of computer memory design, cache concepts, and bus fundamentals (a, b, i, j).
9. Show understanding of the operation of a processor implemented with a microarchitecture (a, j).
10. Demonstrate understanding of processor architecture by writing a portion of an emulator for a processor (b, c, i, k).
11. Show an understanding of interrupts, interrupt handlers, and DMA (b, i).
12. Demonstrate understanding of virtual memory, process concepts, and parallel architectures (b, c, i, j).

Required textbook
Andrew Tanenbaum, *Structured Computer Organization (5th edition)*, Prentice-Hall, 2006. ISBN 0-13-148521-0.

Exams
There will be two 75-minute exams and a final exam (see tentative schedule below). No make-ups will be given unless approved in advance or in case of a valid emergency. In the latter case, contact the instructor immediately.

Projects
There will be several assembly language programming projects to be run on an Intel 8088 simulator together with a Java project to emulate a fictional computer. Each project is due at the beginning of class on the due date. A 10% penalty will be imposed the first day a project is late and an additional 5% penalty for each day thereafter (Monday through Friday only). All project implementations must adhere to the established standards. All project submissions must be your own work and not the result of plagiarism. In particular, group work on projects is not permitted. Violation of this policy will result (at minimum) in a project grade of zero.

Attendance
Attendance is a University requirement, and you are expected to attend every class. Your grade may be adversely affected by any absences. Notify the instructor by telephone or email if you cannot attend.

Grading policy
Projects 25%, 75-minute exams 25% each, final exam 25%. The score used in determining the final grade will be no higher than one letter grade above that derived from tests alone. (Unless curved, A: 90-100, B: 80-89, C: 70-79, D: 60-69, F: below 60.)

Special needs
If you have a disability or acquire one, contact the Director of Services for Students with Disabilities (Walb 113, telephone number 481-6658) for services and accommodations available at IPFW. For more information, visit the web site for SSD at http://www.ipfw.edu/ssd/ .

Tentative schedule and topics

Week of:	Topics	Chapter	Comments
Jan 14	Introduction. Processors. Primary memory.	1.1-1.6, 2.1-2.2	
21	Secondary memory. I/O. Radix number systems and conversion.	2.3-2.4, Appendix A	No class Monday
28	Two's complement. Binary arithmetic. Floating-point numbers.	Appendix A, Appendix B	
Feb 4	Assembly language programming	C.1-C.4	
11	The assembler, tracer, and examples	C.5-C.8	
18	Gates and Boolean algebra. Digital logic circuits.	3.1-3.2	Test 1: Wednesday, Feb 20
25	Memory	3.3	
Mar 3	CPU chips and buses.	3.4-3.5	
10			Spring break week
17	Example buses. Interfacing. An example microarchitecture.	3.6-3.7, 4.1	Last day to withdraw is Friday
24	Example ISA: IJVM. Implementation.	4.2-4.3	
31	Microarchitecture design. Improving performance (cache)	4.4-4.5	Test 2: Wednesday, April 2
April 7	ISA level overview. Data types, instruction formats, & addressing.	5.1-5.4	
14	Instruction types. Flow of control. Traps & interrupts.	5.5-5.6	
21	Virtual memory. Process concepts.	6 (selections)	
28	Parallel architectures	8 (selections)	
May 5	Final Exam: Wednesday, May 7, 6:15 PM – 8:15, ET 109		

ABET Program Outcomes

(These are among the criteria ABET, our accrediting agency, uses in accrediting degree programs in computing.)

a. An ability to apply knowledge of computing and mathematics appropriate to the discipline.
b. An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.
c. An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs.
d. An ability to function effectively on teams to accomplish a common goal.
e. An understanding of professional, ethical, legal, security and social issues and responsibilities.
f. An ability to communicate effectively with a range of audiences.
g. An ability to analyze the local and global impact of computing on individuals, organizations, and society.
h. Recognition of the need for and an ability to engage in continuing professional development.
i. An ability to use current techniques, skills, and tools necessary for computing practice.
j. An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.
k. An ability to apply design and development principles in the construction of software systems of varying complexity.