CS/ECE 347, Spring, 1998
Course Information

1. Organization

Instructors:

Randal E. Bryant
7128 Wean Hall
x8-8821
Randy.Bryant@cs.cmu.edu
M 3:00–4:00

Todd C. Mowry
8123 Wean Hall
x8-3725
tcm@cs.cmu.edu
F 10:00–11:00

Teaching Assistants:

Herb Derby
7130 Wean Hall
x8-8078
derby@cs.cmu.edu
T 1:30–2:30

Jason Flinn
8208 Wean Hall
x8-1425
jflinn@cs.cmu.edu
W 10:30–11:30

Miroslav Velev
2134 Hammerschlag
x8-5241
mvelev@ece.cmu.edu
Th 2:00–3:00

Secretary:
Joan Maddamma
7121 Wean Hall
x8-7656
Joan.Maddamma@cs.cmu.edu

Lecture:
TTh 3:00-4:20, Doherty Hall 2315

Recitations:

A	M	10:30	Hammerschlag B131	Flinn
B	M	12:30	Porter Hall A18A	Derby
C	M	3:30	Porter Hall A18C	Velev

2. Objectives

CS 347 presents the principles, characteristics, and trends of computer systems design at a level appropriate for all computer scientists and computer engineers. It expands on the role of a traditional computer architecture course, focussing largely on the hardware design, to include aspects of the complete system, comprising the hardware, operating system, compilers, and application software.

Our intention is to present material that will benefit all computer scientists, even those who are more concerned with software development than with hardware design. By learning more about what is inside "the box," software developers can make more effective use of computers, often achieving dramatic performance gains.

Students will also gain an understanding of the hardware technology that has fueled the rapid progress of computer systems. Since the technology will continue to change and improve, it is important to see how these changes will affect the characteristics of future computer systems.

After completing the course, students should have gained a firm grounding in the following:

Machine-level programming. With the advent of optimizing compilers, there is seldom need to write large amounts of assembly code. On the other hand, it is important to be able to understand the code generated by these compilers, and to write small sections of code. We will use the Digital Alpha instruction set as our main case study, mostly using assembly code generated by the C compilers.

Instruction Set Design. In addition to Alpha, we will study the general characteristics of other instruction sets, including the Intel x86 and the IBM PowerPC.

Pipeline Processor Implementation. Modern processors are designed to execute instructions in a pipelined, and possibly superscalar fashion. The characteristics of these pipelines and the code scheduling capabilities of an optimizing compiler can greatly affect the performance the processor achieves.

Memory Subsystem. The combination of primary memory, caches, and virtual memory are intended to provide the illusion of high speed access to a large memory space. Understanding how this subsystem operates is critical to efficiently implementing applications with high memory requirements.

Program Optimization. Understanding the characteristics of the machine, as well as the capabilities and limitations of optimizing compilers, application programmers can often tune their programs to achieve much higher performance. Performing this tuning requires mastery of techniques and tools to profile, measure, and benchmark program performance.

I/O and Storage Subsystem. With the advent of multimedia applications, we are seeing stringent, real-time requirements placed on the various I/O and display devices, as well as the busses connecting them to the CPU. Understanding the organization and performance characteristics of this subsystem is therefore critical to current and future software developers.

Parallel Computing. Increasing application performance by using resources operating in parallel has been attempted in a number of different ways for nearly 50 years. Although the track record for these efforts is mixed, there are many indications that parallelism will play a increasingly critical role as the technology and the techniques for exploiting it mature.

3. Textbook

J. L. Hennessy, and D. Patterson, *Computer Organization and Design : The Hardware Software Interface*, Second Edition, Morgan Kaufmann, 1998. [**Note:** Don't try to get by with the previous edition of this book.]

4. Course Requirements

Your participation in the course will involve five forms of activity:

1. Attending the lectures.
2. Participating in the recitations.
3. Homework and laboratory assignments.
4. Reading the text and supplementary handouts.
5. Exams

Attendance will not be taken at the lectures or recitation sections. You will be considered responsible for all material presented at the lectures, much of which is not covered to the same depth in the textbook. In addition, you will find the recitations useful for better understanding the course material, the assignments, and the exam coverage.

There will be two types of assignments in this class. *Homework* assignments will involve solving problems with paper and pencil, and possibly writing and executing small programs. They are intended to reinforce the material covered in the lectures. *Laboratory* assignments will be multi-week efforts providing in-depth understanding of some aspect of computer system design. They will involve some combination of C and assembly programming, as well as performance measurements. For both types of assignments, you will be allowed to work in groups of up to 3 people. Each group will turn in a single report, identifying the group members. We will let you form and regulate your own groups. You are free to work on your own as well.

The textbook covers much of the material taught in the course. In the schedule shown in this document, you will see the reading assignments associated with each section of the course. It would be a good idea to read over each of these assignments twice—a brief reading before the lecture to become familiar with the basic concepts, and a more thorough reading later to understand the details.

5. Policies

5.1. Handing in Assignments

All assignments are due at 3:00 pm on the specified due date. Late assignments will be accepted only if you have arranged for an extension in advance. Such arrangements can be made by contacting one of the instructors, but you should have a good reason for doing so. It is your responsibility to get your assignments done on time. Be sure to work far enough in advance to avoid unexpected problems, such as illness, unreliable or overloaded computer systems, etc.

5.2. Final Grade Assignment

Each student will receive a numeric score for the course, based on a weighted average of the following:

Assignments. The assignments will count a combined total of 50% of your score. The exact weighting of the different assignments will be determined near the end of the course based on our perception of the relative effort required. In any case, each homework will count 3–5%, while each lab will count 8–12% of your score. Since small differences in scores can make the difference between two letter grades, you'll want to make a serious effort on each assignment.

Exams. There will be two in-class exams, each counting 12.5%, plus a final counting 25%.

Grades for the course will be determined by a curve. The total score will be plotted as a histogram, and then approximate cutoff points for the different letter grades will be determined. Individual cases, especially those near the cutoff points may be adjusted upward or downward based on factors such as attendance, class participation, and special circumstances.

5.3. Cheating

All of your assignments allow collaboration, but only with members of your project group. Each assignment must be the sole work of the group turning it in. Assignments will be closely monitored, and students may

be asked to explain any suspicious similarities. The following are guidelines on what collaboration outside of your group is authorized and what is not:

Not Cheating:

- Clarifying ambiguities or vague points in class handouts or textbooks.
- Helping others use the computer systems, networks, compilers, or other system facilities.

Cheating:

- *Sharing code or other electronic files*: either by copying, retyping, looking at or supplying a copy of a file.
- *Sharing written assignments*: Looking at, copying, or supplying a assignment.

Be sure to store your work in protected directories.

The penalty for cheating will depend on the severity of the offense and the student's past record in this regard. *At the very least* the student will be given a score of 0 for the assignment.

6. Class Schedule

Table 1 shows a tentative schedule for the class. The notation "Hi" indicates a homework assignment, while "Li" indicates a lab. Any changes will be announced on the class news group. An updated schedule will be maintained on the class WWW pages.

7. Facilities

7.1. Alpha Processors

This term we will shift from the MIPS processors used in previous versions of this class, and also covered in the book, to Digital Alpha processors. Alpha's have a very simple instruction set and hence form an ideal platform for studying machine-level programming. In addition, Alpha's are among the fastest processors available in the marketplace. We anticipate receiving a shipment of Alpha's later in January, in time for class use by mid-February.

7.2. Electronic Communication

We will use the World-Wide Web (WWW) as the central repository for all information about the class. You can reach this page via the link labeled "course, lecture, and seminar home pages" from the CS home page. The complete URL is:

```
http://www.cs.cmu.edu/afs/cs/academic/class/15347-s98/www/home.html
```

Using the Web, you can:

Class	Date	Day	Topic	Reading	Asst	Lecturer
1	01/13	Tue	Overview	1		Both
2	01/15	Thu	Measurement	2.1–3	H1 Out	TCM
3	01/20	Tue	Integer arithmetic	4.1–7		REB
4	01/22	Thu	Floating Point	4.8	H1 Due, H2 Out	REB
5	01/27	Tue	Fast arithmetic	4.9–12		REB
6	01/29	Thu	Memory Technology	7.1	H2 Due	TCM
7	02/03	Tue	Cache structure	7.2	L1 Out	REB
8	02/05	Thu	Cache performance	7.3		REB
9	02/10	Tue	Virtual memory	7.4, 7.6–9		TCM
10	02/12	Thu	I/O, Storage	8.1–10	L1 Due	TCM
11	02/17	Tue	Exam #1			
12	02/19	Thu	ISA basics	3.1–5, A.1, A.10	H3 Out	TCM
13	02/24	Tue	Procedures	3.6, 3.9–10, A.6		TCM
14	02/26	Thu	Data structures	3.7–8, 3.11	H3 Due, L2 Out	REB
15	03/03	Tue	Code Optimization	Handouts		TCM
16	03/05	Thu	Benchmarking	2.4–9		TCM
17	03/10	Tue	Other ISAs	3.12–15		REB
18	03/12	Thu	Pipelining basics	6.1–3	L2 Due	REB
19	03/17	Tue	Data hazards	6.4–5		REB
20	03/19	Thu	Exam #2			
	03/24	Tue	Spring Break			
	03/26	Thu	Spring Break			
21	03/31	Tue	Control hazards & exceptions	6.6–7	L3 Out	TCM
22	04/02	Thu	Multicycle instructions			TCM
23	04/07	Tue	Superscalar	6.8–6.12		TCM
24	04/09	Thu	Instruction-level parallelism		L3 Milestone	TCM
25	04/14	Tue	Multimedia computing	Handouts		REB
26	04/16	Thu	Network Technology	Handouts	L3 Due, H4 Out	REB
27	04/21	Tue	Internetworking			REB
28	04/23	Thu	Multiprocessors	9.1–10		TCM
29	04/28	Tue	Multiprocessors (cont.)		H4 Due	TCM
30	04/30	Thu	Multiprocessors (cont.)			TCM

Table 1: Tentative Course Schedule

- Obtain copies of any handouts or assignments. This is especially useful if you miss class or you lose your copy.
- Read clarifications and changes made to any assignments, schedules, or policies.
- Find links to any electronic data you need for your assignments

For urgent communication with the teaching staff, it is best to send electronic mail (preferred) or to phone. We have also set up a news group for this class, `cmu.cs.class.cs347`. This group will be used by members of the teaching staff to post announcements and clarifications. You may also post to this group to make queries.