Course Number: CSC 310  
Course Title: Assembly Language Programming and Introduction to Computer Organization  
Number of Credits: 3  
Schedule: Three hours of lecture/discussion per week.  
Prerequisite: a grade of C or better in Engr 356

Catalog Description  

Expanded Description  

Integer representation and arithmetic  
Programmer’s view of computer system  
Load/store architectures and MIPS

C/C++ program translation  
Arithmetic/logic expressions  
If and if-else  
Loops  
Arrays

Stack management and functions  
MIPS register use conventions  
Recursion  
Switch statements  
Multi-dimensional arrays

Floating point representation and arithmetic  
MIPS floating point instructions

Machine language encoding  
Assembly and disassembly

Simple MIPS datapath design  
Single-cycle control  
Multi-cycle control with finite state machine

Interrupts and exceptions  
MIPS exception handling  
Signals in Unix  
Hardware support for interrupts/exceptions

Introduction to x86 assembly language  
Instruction set comparisons
Course Objectives and Role in Program

The objectives of this course include:

- Teach principles of instruction set architecture and assembly language programming
- Teach basic procedures of how a compiler translates C/C++ code to assembly language and perform simple optimizations
- Teach basic principles of interrupt/exception handling
- Explore in detail a simple hardware CPU implementation that supports a small instruction subset; introduce students to computer organization
- Show how C/C++ constructs use hardware resources, and introduce concepts of efficiency and performance below the algorithmic level

Students will translate a number of small C/C++ programs into assembly language, and learn to trace and debug at the assembly level. They will extend the simple CPU implementation introduced in class to support additional instructions. The knowledge of how C/C++ constructs are translated to execute on hardware, simple hardware operations and interrupt handling are crucial building blocks for the Operating Systems and Computer Architecture courses.

Learning Outcomes

At the end of this course students will be able to

- Translate C/C++ code into assembly language
- Perform simple optimizations by hand
- Trace and debug at the assembly level
- Understand and extend simple CPU implementations
- Understand basic interrupt/exception handling
- Make simple performance estimates for assembly code

Method of Evaluation

Student learning will be evaluated on the basis of

- Completeness and quality of programming assignments
- Grade on two quizzes
- Grade on two midterm examinations
- Grade on final examination

The weight assigned to each element of evaluation will be determined by the instructor of the course on the first day of the class.

Required Textbooks

*Computer Organization*, Patterson, D. and Hennessy, J. Morgan-Kaufmann, 2004
*Course notes Vol. 1 and 2*, Hsu, W. 2005
*Lab manual*, Hsu, W. 2005

Modified by: W. Hsu
Last Revision Approved: October 11, 2006