Course Number: CSC 656
Course Title: Computer Organization
Number of Credits: 3
Schedule: Three hours of lecture/discussion per week.
Prerequisite: a grade of C or better in CSc 415 (may be taken concurrently) or consent of instructor

Catalog Description
Instruction set design. Piplelined datapath and control. Cache and memory system design. Input/output subsystems. Software/hardware interactions. Parallel processing. Extra fee required.

Expanded Description

Instruction set architecture
Load/store architecture vs. x86
Simple MIPS datapath/control design

Pipelined datapath and control for integer instructions
Compiler optimizations for pipelines
Exception handling in pipelines
Introduction to multiple instruction issue

Floating point pipelines
Branch prediction and target address delivery

Caches and memory hierarchy
Impact of cache size, block size and associativity on performance
Cache simulations and software interactions
Interactions with virtual memory
Hardware support for operating systems

Input/output subsystem
Interactions with memory system and OS
Performance issues

Course Objectives and Role in Program
The objectives of this course include:

- Teach principles of instruction set architecture
- Teach pipelined datapath and control design
- Overview floating point support and branch handling
- Overview interactions between compiler optimizations, OS, and architecture
- Examine cache and memory hierarchy design
- Study effect of code constructs on memory system behavior and performance
- Overview common input/output technologies and operations

Students will work on projects and assignments involving detailed instruction and memory system traces, to develop good understanding of how software constructs consume hardware resources. Students will make simple extensions to functional units to study the effect of design choices on performance. A strong background in computer organization/architecture is key to doing more advanced work in operating systems and performance modeling/evaluation.

Learning Outcomes
At the end of this course students will be able to

- Understand principles of instruction set architecture
- Understand and extend simple pipeline implementations
- Understand in detail cache and memory system behavior and design
- Estimate performance benefits from compiler optimizations and code transformations
- Develop simple trace-driven simulators for functional units

Method of Evaluation
Student learning will be evaluated on the basis of
• Completeness and quality of assignments and programming projects
• Grade on midterm examination
• 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., Hennessy, J., Morgan-Kaufmann, 2004
Course notes, Hsu, W. 2007

Modified by: W. Hsu
Last Revision Approved: April 21, 2007