

# ECEN/CSCI 5593 - Advanced Computer Architecture

College of Engineering and Applied Science  
University of Colorado - Boulder

Term: Fall 2011

Professor Dan Connors

Meeting: Monday/Wednesday 4:30pm-5:45pm

Email: [dconnors@colorado.edu](mailto:dconnors@colorado.edu)

Course is available for **on-line (distance learning)** through the Center for Advanced Engineering and Technology Education (CAETE) program. Course videos will be available on the web as well as other communication made available for industry students. Students in the CAETE program are afforded delayed deadlines to accommodate work schedules. Information regarding the CAETE program can be found at: <http://cuengineeringonline.colorado.edu/>

Website: Web material (lecture notes, assignments, etc) will be posted using an on-line Moodle system: <https://ecee.colorado.edu/secure/moodle/>

## Course Design

*Catalog Description:* Covers the broad range of concepts, techniques and structures useful in designing modern multicore systems in mobile, general-purpose, data center, and high-performance domains. Topics: Detailed computer design: instruction set architecture (ISA), processor implementation, pipelining, superscalar/out-of-order execution, cache design, and memory hierarchy. Software design: optimizing compiler, run-time systems, operating systems, and parallel programming models. Emerging design trends: fault tolerance, transient faults, defect tolerant design, wearout detection and failure prediction, and power-aware computing.

The primary emphasis of the course is on emerging multicore and GPU (Graphics Processor Unit) architectures: Nvidia Fermi, AMD Fusion APU (Accelerated Processor Units), Intel MIC (Many Integrated Cores).

*Gained Experience:* Students will gain experience exploring advanced architecture concepts:

- CUDA programming for Nvidia GPUs (GTX 480)
- OpenCL (Compute Language) - Intel (Sandy Bridge) and AMD (Fusion) platforms
- OpenMP (Multi-Processing) implementation and examples

In addition, students will utilize binary instrumentation infrastructure (PIN), hardware performance event counters, and advanced architecture simulators; and examine architecture details of ARM, IBM Cell, Network-on-Chip (NOC) Tiler, etc.

*Overview and Objectives:* Explore advanced concepts and state-of-the-art developments in computer architecture: memory systems, multicore processors, simultaneous multithreading, run-time optimization, fault tolerance, system reliability, power-aware computing, and embedded/mobile computing. This course presents the principles, characteristics, and trends of computer systems design at a level appropriate for all computer scientists and computer engineers. Participants will be able to recognize the relationships between computer design, application & computing domain requirements, cost/performance tradeoffs, and business trends in the computer industry. Students will also gain an understanding of the historical hardware and software technology that has fueled the rapid progress of computer systems.

## **Course Policies:**

### **Reference textbook:**

*Computer Architecture: A Quantitative Approach* by Hennessy and Patterson, 4th Edition - Morgan Kaufmann, ISBN: 0123704901

Lecture: Lecture material (slides and notes) will be made available on the web prior to class. Lecture will also consist of chalk drawings, overhead drawings, and content not explicitly present in slides and notes.

### **Assessment Designs**

(30 %) Assignments (Checkpoints)

(30 %) Final Exam

(30 %) Final Project

(10 %) Paper reviews and class participation

Assignments: Homework assignments (called Checkpoints) are designed to give students opportunities to study characteristics of architecture concepts using experimental infrastructure (simulators, multicore systems, benchmarks, etc). Students will have access to multicore and GPU resources for the checkpoints.

Final Exam: Examinations are intended to measure your individual mastery of the material. The exams will generally test your knowledge of assignment material, so you are responsible for mastering all lab, homework, and programming material submitted with other partners, as if you did all the work by yourself. All exams will be open book and open notes (unless otherwise stated). The nature of the course material is such that the final exam must be cumulative.

Final Project: Students will have opportunities to work together, individually, or on a survey paper. The final project consists of an exploration of a novel idea within the context of computer architecture research. Examples include new methods of branch prediction, cache prefetching, or programming a task/algorithm/kernel in a parallel language or language extension.

Paper Reviews and Participation: An element of studying advanced architecture includes reviewing current publications on course topics. Students will write 1-page reviews of designated publications. Class discussions and participation are essential components of this course.

**Course Policies:** Policies regarding class attendance, turning in late work, missing homework, tests or exams, make-ups, requesting extensions, reporting illnesses, cheating and plagiarism, changes to the syllabus. Academic policies will be consistent with the University's policies at the College of Engineering and Applied Science's

Extensions/make-ups: In general, late work will not be accepted. Turn in all work by the established deadline. In case you have difficulties finishing an assignment contact the instructor before the deadline. Late work can be accepted only under circumstances beyond student's control and after arrangement with the Instructor, prior to the deadline. You have to follow the submission and policies and guidelines published on the course moodle, email is not a good way to submit assignments. Plagiarism is the passing of someone else's work as one's own, without giving the original author due credit. Scholastic dishonesty will be treated very strictly as per University of Colorado rules.

**Course Resources:** Students will be granted access to multicore and GPU resource servers. ECE Majors, ECE Graduate Students, and students currently registered for courses which require access to ECE computer labs are eligible to obtain an ECES account. Unfortunately, no outside university support will be offered to install the software on the student’s home system.

For more support (tutorials, information), go to: <http://eces.colorado.edu/getstarted.html>  
 To create your ECES account, please visit: [sac.colorado.edu](http://sac.colorado.edu)

### Course Topics & Tentative Schedule

Concepts
Design issues of architectures (parallel architectures and fundamental design issues), technology (scaling, nanotechnology, 3D die stacking).
Modern Processor Design: pipelining, branch prediction, superscalar/out-of-order execution, predicated execution, cache design, etc.
Compiler Optimization: ILP and profile-guided optimization: loop unrolling, inlining, superblock formation, etc.
Run-time Systems: Feedback-directed Optimization (FDO), dynamic optimization, binary instrumentation/translation, hardware performance event counters, and auto-tuning.
Fault Tolerance and Reliability Design: transient faults, defect tolerant design, wearout detection, and failure prediction.
Power-aware Computing: (static/dynamic power, tri-gate transistor technology, DVFS-Dynamic Voltage Frequency Scaling), and energy-efficient computing and storage.
Multicore and GPU Architectures: Nvidia Fermi, AMD APU, Intel MIC, etc.), and other systems: Network-on-Chip, IBM Cell, etc.
Nvidia CUDA (Compute Unified Device Architecture): CUDA programming model, tools, languages, and libraries for GPU computing Advanced CUDA: optimization, irregular parallelism.
OpenCL – Open Compute Language: Program compilation and kernel objects, Managing buffers, Kernel execution, Kernel programming – basics, Kernel programming – synchronization.
Parallel Programming Support – Algorithms, languages, tools, debugging, profiling, compilers, and libraries. OpenMP overview.

### Instructor Background

Dr. Dan Connors is a veteran of the computer architecture and scientific computing field. He received his Ph.D. in Computer Engineering from the University of Illinois at Urbana-Champaign where he worked in the IMPACT compiler group under Professor Wen-mei Hwu. For his commitment to teaching, Dr. Connors was awarded the University of Colorado College of Engineering Peebles Outstanding Teaching Award in 2003 and the University of Colorado College of Engineering Sullivan-Carlson Teaching Innovation Award in 2008. Dr. Connors serves as a leading industry consultant for the development of education infrastructure for high-performance systems and parallel programming initiatives for emerging applications. As an early adopter of NVIDIA’s CUDA programming model, Dr. Connors has delivered industry courses to Intel, Hewlett-Packard, Lockheed-Martin, and Northrop-Grumman. For his work in CUDA development, Dr. Connors received an NVIDIA Faculty Partnership Award.