# EE 5123 --- Computer Architecture (Spring 2017)
# Course Syllabus

**Instructor**:
Lide Duan
Assistant Professor
Department of Electrical and Computer Engineering
University of Texas at San Antonio
http://engineering.utsa.edu/~lideduan/index.html

**Office**: AET 2.376
**Phone**: 210-458-5208
**Email**: lide.duan@utsa.edu
**Class Meeting Time:** 6:00-7:15PM (Mondays and Wednesdays)
**Class Location**: EB 2.04.04
**Office Hours**: 4:00 – 6:00PM (Mondays and Wednesdays) or by appointment.
**Website**: All class materials will be handled electronically through Blackboard Learn.

**Course Description**:
Computer architecture fundamentals, instruction set architectures, memory and cache hierarchy, microprocessor pipelining, instruction-level parallelism, data-level parallelism, thread-level parallelism, and request-level parallelism.

**Prerequisite**:
Graduate standing. Non-experience in computer architecture is assumed. However, course projects will require programming skills in high-level languages (e.g., C, C++, Java, Python).

**Textbook**:
Computer Architecture: A Quantitative Approach, 5th Edition. (By John Hennessy and David Patterson).

**Course Topics** (tentative):
- Fundamentals of computer architecture
- Instruction set architectures (ISA)
- Cache and memory hierarchy design
- Pipelining
- Instruction-level parallelism (ILP)
    - Branch predictions, dynamic scheduling, multiple issue and static scheduling, speculations, compiler techniques, ILP limitations, etc.
- Data-level parallelism (DLP)
    - Vector architectures, SIMD, GPUs, loop-level parallelism.
- Thread-level parallelism (TLP)
    - Centralized shared-memory, distributed shared-memory, multiprocessor cache coherence, synchronization, consistency, etc.

**Grading Policy**:
- Homework assignments: **30%**
    - 3 assignments of 10% each

- Exams: **35%**
  - Exam 1 (10%) + Exam 2 (10%) + Final (15%)
- Project: **30%**
  - Project 1 (15%) + Project 2 (15%)
- Quizzes: **5%**
  - The instructor will randomly choose a few classes to ask the students to turn in an answer to a very simple question. These are mainly for checking attendance.
- Total: **100%**

**About the Grading**

- The final letter grades will be curved based on the ranks and score gaps.
- After the grade of each assignment/exam/project/quiz is posted, you have up to a week to see me for any misgrading or miscalculation. After that, the grade is finalized.

**About the HW Assignments**

- Tentatively, the 3 assignments will be out in the 4th, 9th, and 15th week of the semester, and due in about one week.
- All must be turned in through Blackboard Learn (in pdf) before the specified deadlines. No late turn-ins.
- HW questions are good examples of exam questions.

**About the Exams**

- Tentatively, Exam 1 will be on Monday 2/13 (class time); Exam 2 will be on Monday 3/27 (class time); and Final Exam will be on Monday 5/8 (6PM).
- All exams are open-books and open-notes. You can bring your laptop for accessing class materials, but Internet access and communication with other are not allowed.
- No make-up exams (except for extremely special situations with legitimate proof and under discretion of the instructor).

**About the Projects**:

- There will be two projects. Tentatively, Project 1 will be assigned in the 3rd week and due in the 9th week (before Spring Break); Project 2 will be assigned in the 11th week (after Spring Break) and due in the 16th week (last week of the semester).
- Project 1 is a disassembler of MIPS binary code. You will be given a MIPS binary file (in txt format), and need to disassemble it into the corresponding MIPS assembly code.
- For Project 2, a cache performance simulator will be assigned. However, you will have a choice to either implement it or submit a term paper instead. The term paper should be based on a number of research articles on a specific topic in computer architecture.
- The projects can be implemented in any high-level language (e.g., C, C++, Java, Python, etc.) under any operating system (e.g., Linux, Mac OS, Windows, etc.). However, you should use a free environment in order for the instructor to reproduce your results. Therefore, developing under Linux using free compilers is highly recommended. In your submission, you will need to include a readme file (in txt) that specifies the instructions to compile and run your program. It is your responsibility to make sure that your program can be run successfully by following your readme file with reasonable effort.
- **Both projects must be completed individually**. Discussions with others are allowed, but copying source code (from either your classmates or students who took this course before) is NOT! Automatic source code checking tools, in addition to manual effort, will be applied to check for coding plagiarism. All parties involved (either copying or being copied) will lose all the points for the project. So, protect your source code!