

CSE 3120 Computer Architecture and Assembly Programming (3 credits)

Primary instructor: William Allen

Supporting instructor: Shengzhi Zhang

Textbooks and references:

Kip R. Irvine, Assembly Language for x86 Processors, Prentice Hall, 7th Edition, 2015.
(R)

Course information:

2014–2015 Catalog description: CSE 3120 Computer Architecture and Assembly Programming (3 credits) Introduces advanced computer architecture concepts. Includes microcode, execution pipelines, cache management, vector processors, parallel architectures and RISC processors. Explores the interfacing of assembly language programs with the operating system and high-level languages. Requires students to interface assembly with C and the Win32 API. Prerequisites: CSE 2050, CSE 2120.

Prerequisites by topic: Computer organization and programming in a systems-level language

Place in program:

Computer Science Program: Required

Software Engineering Program: Advanced CSE elective.

Course outcomes & related student outcomes: The student will be able to

1. Relate architectural concepts to program design and performance, such as cache memory, microprogramming, instruction pipelines, multiprocessing, and embedded systems. (1: Fundamental knowledge)
2. Implement macros, stack frames, modular programming, dynamic memory allocation and recursion in assembly language. (3: Skillful use of tools)
3. Use programming techniques for graphics, user interfaces, and file I/O. (4a: Skillful software construction)
4. Reverse engineering in-line assembly and advanced debugging techniques. (1: Fundamental knowledge)
5. Explain the application of computer architecture and assembly programming topics to fields, such as security, operating system design, and databases. (2: Scientific, computing, and engineering problem solving)

Topics covered:

1. Introduction, overview of topics for this course (1.5 hours)
2. Review: Boolean and digital logic, and computer organization (4.5 hours)
3. More digital circuits: Shift registers, counters, and memory (1.5 hours)

4. Virtual memory, page tables, caches, etc. (3.0 hours)
5. Instruction sets, architecture design goals, and instruction encoding (1.5 hours)
6. Microprogrammed control and microcode (1.5 hours)
7. Pipeline details, hazards, and performance (1.5 hours)
8. Parallel and distributed processing (3.0 hours)
9. Macros and macro expansion, dynamic memory allocation, and recursion (3.0 hours)
10. Procedure calls, stack frames, and parameter passing (6.0 hours)
11. BIOS calls, 16-bit assembly code, self-modifying code, and applications in embedded computing/security (1.5 hours)
12. Assemblers, compilers, and debuggers (1.5 hours)
13. Data structures: Arrays, strings, and unions (3.0 hours)
14. Linking, object formats, and executable files (1.5 hours)
15. Interfacing assembly language to Win32 and other APIs (3.0 hours)
16. In-line assembly (1.5 hours)
17. Interfacing assembly language to high-level languages (3.0 hours)
18. Examinations (1 each on Organization and Assembly) (3.0 hours)

Approved by: William Allen, Associate Professor & Shengzhi Zhang, Assistant Professor

Signature:  Date: 2/28/2015

Signature:  Date: 2/2/2015