Revised 8/22

NOVA COLLEGE-WIDE COURSE CONTENT SUMMARY
CSC 215 – COMPUTER SYSTEMS (3 CR.)

Course Description

Examines the hierarchical structure of computer systems. Explores the representation of instructions and data, memory organization/structure, structure of a CPU, programming hierarchy and operating system interactions. Lecture 3 hours.

General Course Purpose

CSC 205 or CSC 215 is intended to fulfill a first course in Computer Architecture, Organization and Systems in the CS curriculum. The focus of CSC 215 is on Systems with a sampling of Architecture and Organization content.

Course Prerequisites/Corequisites

Prerequisite: CSC 221.

Course Objectives

Upon completing the course, the student will be able to:

- Machine level data representations
    - Describe how numbers, text, and other analog or discrete information can be represented in digital form
    - Interpret computer data representation of unsigned integer, signed integer (in 2's complement form) and floating-point values in the IEEE-754 formats
    - Explain the impact due to the limitations of data representations such as rounding effects and their propagation affect the accuracy of chained calculations, overflow errors, and mapping of continuous information to discrete representation
- CPU and Instruction Set Architecture
    - Differentiate various instruction set architectures
- Memory Hierarchy
    - Identify the memory technologies found in a computer processor and computing systems
    - Describe the various ways of organizing memory and the impacts on cost-performance tradeoffs, speed, capacity, latency, and volatility (also include long term storage with tape drives, hard drives, and SSDs with performance enhancements like RAID)
    - Describe the operation of common cache mapping schemes: Direct, Associative, and Set associative
- Digital Logic, Digital Systems, and Digital Design
    - Design a simple combinational circuit using logic gates
    - Apply Boolean functions, algebraic theorems, and Karnaugh Maps to simplify combinational circuits
- Memory model
    - Explain how high-level languages structure memory into stack, static, and dynamic regions and explain how each is used to include mapping logical addresses to physical memory chips
- Virtual Memory
    - Examine the hardware and control structures that support virtual memory
    - Explain how logical addresses are mapped to physical addresses by the OS
    - Explain how VM is used for caching and memory protection
- Security
    - Explain how security concerns impact the development and design of computer systems

- Operating Systems
    - Explain the general structure of a multi-programmed operating system
    - Explain how a common file system works, including structure, I/O operations, and security
    - Describe the role of the kernel in operating systems
    - Compare processes vs. threads in terms of how each are created, what resources are shared, and how they communicate
- Language hierarchy
    - Explain how programming language abstractions at multiple levels are translated to lower levels and executed
- C/Linux
    - Complete C programming projects on the command line using common Linux utilities (e.g., cd, ls, pwd, mkdir, rmdir, rm, cat, cp, man, tar, nano)
    - Compile and run C programs using gcc and makefiles
    - Debug C programs using gdb to step through code, stop at breakpoints, and examine variables/registers/memory
    - Write C programs that perform low-level manipulations involving bitwise operations, masking, memory manipulation and management, structs and unions, signed vs. unsigned integers, strings, arrays, and file I/O

Major Topics to be Included

- Machine level data representations
- CPU and Instruction Set Architecture
- Memory Hierarchy
- Digital Logic, Digital Systems, and Digital Design
- Memory model
- Virtual Memory
- Security
- Operating Systems
- Language hierarchy
- C/Linux