CS 35101   COMPUTER ARCHITECTURE    3 Credit Hours

Contributor’s Name: Robert Walker
Other Instructors Consulted: Dianne Foreback
   (Drs. Kim & Sharma were listed but have not taught the course)

Patterson & Hennessy, Morgan Kaufmann 2014

Course Content (current catalog):
   Functional overview of computer systems, interconnection of basic components, system performance measures, instruction set design, arithmetic logic unit, control unit, memory system, pipelining, interrupts and input-output.

Prerequisites or co-requisites: CS 13001

Required, elective, or selected elective: Required for both BS and BA

Course Learning Outcomes:
   • The role of performance measurement and evaluation
   • The basics of an instruction set and assembly language programming using RISC architecture
   • The important principles of computer organization and key ideas such as representation of integers and floating-points numbers, and arithmetic algorithms
   • The key ideas in control mechanisms including pipelining
   • The organization and control
   • Principles of memory hierarchies and unifying the ideas of caching and virtual memory
   • The key principles of multiprocessors and multicomputer systems

Topics to be Covered (45-hour BDS content):

   2  Computer Abstractions and Technology
   4  Representing Integers and Characters
   15 Instructions: Language of the Computer
   4  Performance and Basics of Logic Design
   2  Multiplication and Division for Computers
   3  Floating Point
   9  Processor
   3  Memory
   3  Additional Depth and/or Current Topics
Applicable ABET Student Outcomes (SOs): ???

COMMENTS FOR CURRICULUM COMMITTEE
The “Course Learning Outcomes” and list of BDS Topics represents the what Professor Foreback is currently teaching in the course. Other faculty may choose to cover some topics in more depth, and/or to cover an additional topic, so the topic “Additional Depth and/or Current Topics” permits that freedom.
Learning Outcomes to be Covered (ACM CS2013 content):

AR/Digital Logic and Digital Systems
1. Describe the progression of computer technology components from vacuum tubes to VLSI, from mainframe computer architectures to the organization of warehouse-scale computers. [Familiarity]
2. Comprehend the trend of modern computer architectures towards multi-core and that parallelism is inherent in all hardware systems. [Familiarity]
3. Explain the implications of the “power wall” in terms of further processor performance improvements and the drive towards harnessing parallelism. [Familiarity]
4. Articulate that there are many equivalent representations of computer functionality, including logical expressions and gates, and be able to use mathematical expressions to describe the functions of simple combinational and sequential circuits. [Familiarity]
5. Design the basic building blocks of a computer: arithmetic-logic unit (gate-level), registers (gate-level), central processing unit (register transfer-level), memory (register transfer-level). [Usage]
6. Use CAD tools for capture, synthesis, and simulation to evaluate simple building blocks (e.g., arithmetic-logic unit, registers, movement between registers) of a simple computer design. [Usage]
7. Evaluate the functional and timing diagram behavior of a simple processor implemented at the logic circuit level. [Assessment]

AR/Machine Level Representation of Data
1. Explain why everything is data, including instructions, in computers. [Familiarity]
2. Explain the reasons for using alternative formats to represent numerical data. [Familiarity]
3. Describe how negative integers are stored in sign-magnitude and two's-complement representations. [Familiarity]
4. Explain how fixed-length number representations affect accuracy and precision. [Familiarity]
5. Describe the internal representation of non-numeric data, such as characters, strings, records, and arrays. [Familiarity]
6. Convert numerical data from one format to another. [Usage]
7. Write simple programs at the assembly/machine level for string processing and manipulation. [Usage]

AR/Assembly Level Machine Organization
1. Explain the organization of the classical von Neumann machine and its major functional units. [Familiarity]
2. Describe how an instruction is executed in a classical von Neumann machine, with extensions for threads, multiprocessor synchronization, and SIMD execution. [Familiarity]
3. Describe instruction level parallelism and hazards, and how they are managed in typical processor pipelines. [Familiarity]
4. Summarize how instructions are represented at both the machine level and in the context of a symbolic assembler. [Familiarity]
5. Demonstrate how to map between high-level language patterns into assembly/machine language notations. [Familiarity]
6. Explain different instruction formats, such as addresses per instruction and variable length vs. fixed length formats. [Familiarity]
7. Explain how subroutine calls are handled at the assembly level. [Familiarity]
8. Explain the basic concepts of interrupts and I/O operations. [Familiarity]
9. Write simple assembly language program segments. [Usage]
10. Show how fundamental high-level programming constructs are implemented at the machine-language level. [Usage]

AR/Memory System Organization and Architecture
1. Identify the main types of memory technology (e.g., SRAM, DRAM, Flash, magnetic disk) and their relative cost and performance. [Familiarity]
2. Explain the effect of memory latency on running time. [Familiarity]
3. Describe how the use of memory hierarchy (cache, virtual memory) is used to reduce the effective memory latency. [Familiarity]
4. Describe the principles of memory management. [Familiarity]
5. Explain the workings of a system with virtual memory management. [Familiarity]
6. Compute Average Memory Access Time under a variety of cache and memory configurations and mixes of instruction and data references. [Usage]

AR/Interfacing and Communication
1. Explain how interrupts are used to implement I/O control and data transfers. [Familiarity]
2. Identify various types of buses in a computer system. [Familiarity]
3. Describe data access from a magnetic disk drive. [Familiarity]
4. Compare common network organizations, such as ethernet/bus, ring, switched vs. routed. [Familiarity]
5. Identify the cross-layer interfaces needed for multimedia access and presentation, from image fetch from remote storage, through transport over a communications network, to staging into local memory, and final presentation to a graphical display. [Familiarity]
6. Describe the advantages and limitations of RAID architectures. [Familiarity]

AR/Functional Organization
1. Compare alternative implementation of datapaths. [Familiarity]
2. Discuss the concept of control points and the generation of control signals using hardwired or microprogrammed implementations. [Familiarity]
3. Explain basic instruction level parallelism using pipelining and the major hazards that may occur. [Familiarity]
4. Design and implement a complete processor, including datapath and control. [Usage]
5. Determine, for a given processor and memory system implementation, the average cycles per instruction. [Assessment]
AR/Multiprocessing and Alternative Architectures
1. Discuss the concept of parallel processing beyond the classical von Neumann model. [Familiarity]
2. Describe alternative parallel architectures such as SIMD and MIMD. [Familiarity]
3. Explain the concept of interconnection networks and characterize different approaches. [Familiarity]
4. Discuss the special concerns that multiprocessing systems present with respect to memory management and describe how these are addressed. [Familiarity]
5. Describe the differences between memory backplane, processor memory interconnect, and remote memory via networks, their implications for access latency and impact on program performance. [Familiarity]

AR/Performance Enhancements
1. Describe superscalar architectures and their advantages. [Familiarity]
2. Explain the concept of branch prediction and its utility. [Familiarity]
3. Characterize the costs and benefits of prefetching. [Familiarity]
4. Explain speculative execution and identify the conditions that justify it. [Familiarity]
5. Discuss the performance advantages that multithreading offered in an architecture along with the factors that make it difficult to derive maximum benefits from this approach. [Familiarity]
6. Describe the relevance of scalability to performance. [Familiarity]