

## COURSE DESCRIPTION

Dept., Number	CS470	Course Title	Computer Architecture
Semester hours	3	Course Coordinator	Virgil Bistriceanu, Instructor

### Current Catalog Description

Introduction to the functional elements and structures of digital computers. Detailed study of specific machines at the register transfer level illustrates arithmetic, memory, I/O, and instruction processing. Prerequisites: CS 350 and ECE 218. (2-2-3) (T) (C)

### Textbook

Computer Organization and Design: the hardware/software interface, David A. Patterson, John L. Hennessy, edition 3/e, Morgan Kaufmann, Inc. ISBN-10: 0123706068, 2005

### References

See [www.cs.iit.edu/~cs470](http://www.cs.iit.edu/~cs470)

### Course Outcomes

Students should be able to:

- Present the milestones of computer architecture history
- Fundamentals of computer design
  - Explain the difference between various measure of performance: Latency, throughput; MIPS, MPFLOS
  - Comparing performance
  - Utilize Amdahl's law to estimate the overall speedup
  - Explain the difference between a good and a bad benchmark
- Assembly level machine organization
  - Explain the basic organization of the classical von Neumann machine and its major functional units
  - Explain how an instruction is executed in a classical von Neumann machine
  - Summarize how instructions are represented at both the machine level and in the context of a symbolic assembler
  - Explain different Instruction Set formats (0 (stack), 1 (accumulator), 2, and 3-addresses per instruction; Variable length vs. fixed length formats)
  - Design the Instruction Set for a general purpose CPU
  - Explain how the basic addressing modes work: Register, Memory direct, Memory indirect, Base and displacement, Indexed
  - Explain how base and displacement addressing is used in block-based

- programming languages
- Write small MIPS assembly language programs
- Demonstrate how fundamental high-level programming constructs are implemented at the machine-language level: If-then-else, Loops (for, while, do-until), Procedure call/return
- Explain the basic concepts of interrupts and I/O operations
- Datapath and Control
  - Design a single clock-cycle datapath for a CPU
  - Explain why a single clock-cycle datapath is inefficient
  - Re-factor a single clock-cycle datapath into a multi clock-cycle one
  - Explain the difference between a hardwired and a microprogrammed control unit
  - Design the control unit for a single clock-cycle datapath
  - Explain how exceptions impact the design and performance of a datapath
- Pipelining
  - Derive the formula for the throughput of an ideal pipeline with N stages
  - Explain the limiting factors in building a pipeline with too many stages
  - Explain how data and control hazards occur and how their impact can be eliminated or reduced
  - Re-factor MIPS code to reduce/eliminate data and branch hazards
  - Explain the significance of a late commit in the pipeline
  - Explain the changes in the design and implementation of a pipelined datapath to account for exceptions
  - Explain branch prediction
  - Solve problems that require finding the *real* CPI of a program running on a pipelined datapath
- The memory hierarchy
  - Identify the main types of memory technology and explain the trade-off in using them
  - Explain the effect of memory latency on running time
  - Explain the use of memory hierarchy to reduce the effective memory latency
  - Explain the differences between different cache organizations: Direct mapped, Set associative Fully associative
  - Utilize a cache simulator and access traces to compare the performance of caches with different sizes and organizations
  - Explain main memory organization alternatives to improve performance: Wide-memory, Interleaving
  - Explain the impact of access stride to performance
  - Explain the virtual memory structure and mapping
  - Explain why and how virtual memory impacts performance and how performance can be improved. TLB
  - Analyze the differences between cache organizations in systems with virtual memory: Real address caches, Pipelined real caches, Virtual address cache, Restricted virtual caches, TLB addressing
- I/O
  - Define the meaning of various I/O performance measures
  - Types and characteristics of I/O devices
  - Explain the differences between major buses (IDE, SCSI, USB, PCI):

synchronous v. asynchronous, Serial v. parallel, Number of devices, Termination, Transfer rates

- Design issues related to I/O system addressing: Memory-mapped I/O, Cache coherency, Snoopy controllers, DMA I/O configurations
- Explain the sources of latency in a I/O subsystem

#### Relationship between Course Outcomes and Program Outcomes

The following Program Outcomes are supported by the above Course Outcomes:

- a. An ability to apply knowledge of computing and mathematics appropriate to the discipline
- c. An ability to design, implement and evaluate a computer-based system, process, component, or program to meet desired needs
- f. An ability to communicate effectively with a range of audiences.
- j. An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices

#### Prerequisites by Topic

Basic understanding of a von-Neumann computer organization

The ability to explain the differences between a high level instruction and a compiled instruction

Knowledge of the steps involved in the execution of an instruction

Solid understanding of basic building blocks for a datapath: ALU, register, counter, multiplexer, decoder, glue logic

Working knowledge of Boolean logic

#### Major Topics Covered in the Course

1. Overview and history of computer architecture	1 hour
2. Fundamentals of computer design	3 hours
3. Basic organization of a von Neumann computer	1 hours
4. Instruction Set design	3 hours
5. Datapath and Control	4 hours
6. Pipelining	5 hours
7. The memory hierarchy	4 hours
8. I/O	4 hours
Introduction: discuss class structure, objectives, and requirements, Midterm	3 hours
Project presentation	2 hours
Laboratory	30 hours
Final Exam	-
	Total 60 hours

#### Assessment Plan for the Course

End of every semester Course Objective Assessments by CS department. End of semester Course Evaluations by IIT. Reviewed every Spring semester by CS Undergraduate Studies Committee for possible updates in the following Fall. Once every 4-5 years a detailed review of all materials for the course is made by the CS Undergraduate Studies Committee.

How Data in the Course is Used to Assess Program Outcomes (unless adequately covered already in the assessment discussion under Criterion 4)

See the assessment discussion under Criterion 4

*For a computer science program*

Estimate Curriculum Category Content (Semester hours)

Area	Core	Advanced	Area	Core	Advanced
Algorithms		1.5	Software design		
Data structures			Concepts of programming languages		1.5