In the Name of God
Sharif University of Technology
Department of Computer Engineering

Course Descriptions

B.Sc of Computer Engineering at Sharif University of Technology

Introduction

This document contains the descriptions of the courses offered in the Bachelor of Science degree in Computer Engineering at Sharif University of Technology. The source of the course descriptions is docs.ce.sharif.edu, which is available only in Persian. It is recommended that the auto-translate feature of a web browser (Edge or Chrome) be used for easy understanding of the content, if the viewer intends to verify this document themselves.
This document includes the content of 64 courses. The courses are brought in no particular order.
This document also contains Graduate level courses that are popular among Bachelor students at Sharif's CE Department. It is noteworthy that a Graduate level course does count towards the progression of the B.Sc. degree.

SHARIF UNIVERSITY OF TECHNOLOGY
Department of Computer Engineering

M. A. Abam
VC for education
computer engineering department
sharif university

Address: P.O.Box 11155-4517, Azadi Ave., Tehran, Islamic Republic of IRAN
Telephone: (98)-(21)66166600, Fax: (98)-(21)66019246

Course Descriptions

B.Sc of Computer Engineering at Sharif University of Technology

Introduction

This document contains the descriptions of the courses offered in the Bachelor of Science degree in Computer Engineering at Sharif University of Technology. The source of the course descriptions is docs.ce.sharif.edu, which is available only in Persian. It is recommended that the auto-translate feature of a web browser (Edge or Chrome) be used for easy understanding of the content, if the viewer intends to verify this document themselves.
This document includes the content of 64 courses. The courses are brought in no particular order.
This document also contains Graduate level courses that are popular among Bachelor students at Sharif's CE Department. It is noteworthy that a Graduate level course does count towards the progression of the B.Sc. degree.

Contents

1 System-on-Chip Design

Course Code: 40757, Units: 3, Level: Master's, Prerequisite: None

Course Summary:

With the advancement of integrated circuit technology, it is now possible to integrate various components of a digital system, including processors, memory, digital and analog blocks, and communication blocks, onto a single chip. This is referred to as System-on-Chip (SoC). In this course, students will learn how to design SoCs and will cover important concepts such as co-design of hardware and software, platform-based design, multi-processor SoCs, on-chip interconnection networks, and testing of SoCs.

Course Outline:

- Introduction to SoC architecture and design topics
- Co-design of hardware and software
- Design for platform-based systems
- Multi-processor SoCs (MPSoC)
- On-chip interconnection networks
- SoC testing (digital logic cores, embedded memory, analog cores, mixed-signal testing)

Books Used:

- Michael J. Flynn, Wayne Luk, *Computer System Design: System-on-Chip*, John Wiley & Sons, Inc., 2011.
- Laung-Terng Wang, Charles E. Stroud, Nur A. Touba, *System-on-Chip Test Architectures*, Morgan Kaufmann Publishers, 2008.
- Natalie Enright Jerger, Tushar Krishna, and Li-Shiuan Peh, *On-Chip Networks*, Second Edition, Morgan & Claypool Publishers, 2017.

Evaluation:

- Theoretical Exercises: 3 points
- Mid-term and Final Exams: 15 points
- Quizzes: 2 points

2 Embedded Systems

Course Code: 40462, Units: 3, Prerequisite: Computer Architecture

Course Summary:

An embedded system is a computer system designed to perform specific tasks within a larger system, which is typically non-computational. Statistics show that the majority of computers worldwide (over 80%) are embedded systems. Embedded systems are also fundamental to concepts in computer engineering, such as cyber-physical systems and the Internet of Things (IoT). This course introduces the design and analysis of embedded systems, with a focus on microcontrollers, system architectures, hardware and software structure, and programming techniques for embedded applications.

Course Outline:

- Introduction to Embedded Systems (1 session)
  - Overview of embedded systems, their importance, and applications
  - Reactive systems, sensors, and actuators
- Microcontrollers as Key Elements of Embedded Systems (2 sessions)
  - Introduction to microcontrollers, their architecture, and importance
  - Analog-to-digital conversion, ports, and serial communication
- Resource and Task Management in Microcontrollers (3 sessions)
  - Software implementation of embedded systems
  - Role of infinite loops, remote debugging, and emulators
- Common Hardware Platforms for Embedded Systems (6 sessions)
  - Overview of Arduino and Raspberry Pi as common embedded platforms
  - Comparison and application areas of these platforms
- Automata-Based Programming (4 sessions)
  - Introduction to automata-based programming for reactive systems
  - Hierarchical structures and Mealy/Moore descriptions

- StateCharts Language (3 sessions)
  - MoC in StateCharts and hierarchical design
  - Event and reaction descriptions, use of timers, and real-time concerns
- Real-Time, Energy Consumption, and Reliability in Embedded Systems (4 sessions)
  - Methods for reducing energy consumption and improving reliability
  - Trade-offs between real-time performance, energy consumption, and reliability
- Introduction to the Internet of Things and Embedded Systems Role (7 sessions)
  - IoT definitions, applications, and architectures
  - Communications challenges, D2D communication, 5G, and energy concerns in IoT

Books Used:

- Peter Marwedel, *Embedded System Design*, 1st Edition, Springer, 2006.
- Ahmad Kardaan and Seyed Amir Asghari, *Applications of Embedded Systems in Measurement and Control*, Kian Rayaneh Sabz, 2008.
- Adrian McEwen and Hakim Cassimally, *Designing the Internet of Things*, 1st Edition, Wiley, 2013.
- Online documents on Raspberry Pi and Arduino.

Evaluation:

- Exercises: 3 points
- Project: 2 points
- Mid-term and Final Exams: 13 points
- Quizzes: 2 bonus points

3 Real-Time Systems

Course Code: 40453, Units: 3, Prerequisite: None, Co-requisite: Operating Systems

Course Summary:

The goal of this course is to introduce students to the concepts of real-time systems and timely task execution. The students will learn to design and analyze systems that not only compute the correct results but also perform tasks on time despite the presence of various periodic and non-periodic tasks. Understanding the conditions for real-time feasibility, proper task scheduling, and prioritization is also a key objective of this course.

Course Outline:

- Introduction to Real-Time Systems, Classifications, and Applications
  - Motivation, definitions, types of tasks (soft vs. hard, periodic vs. aperiodic), components of a typical real-time system
- Modeling and Verification of Real-Time System Characteristics (using Petri Nets)
  - Concurrent processing, resource sharing, liveness and boundedness, deadline and timing constraints, execution time estimation and analysis
- Periodic Task Scheduling on a Single Processor
  - Static and dynamic priority scheduling (Rate Monotonic, EDF, etc.)
  - Schedulability analysis criteria
- Non-preemptive vs. Preemptive Tasks
- Scheduling of Aperiodic and Sporadic Tasks in Combination with Periodic Tasks on a Single Processor
  - Schedulability conditions and task distribution methods
- Scheduling Algorithms (FCFS, Polling Server, Deferred Server, Slack Stealing, Sporadic Servers, etc.)
- Brief Overview of Multi-Processor Task Scheduling

- Reliability, Availability, and Fault Tolerance in Real-Time Systems
- Real-Time Communications and Protocols
  - Time constraints in communication and real-time protocols in networks

Books Used:

- G. C. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*, 3rd Edition, Springer, 2011.
- J. W. S. Liu, *Real-Time Systems*, Prentice Hall, 2000.
- Ph. A. Laplante, *Real-Time Systems Design and Analysis*, 3rd Edition, IEEE Press & Wiley InterScience, 2004.
- C. M. Krishna and Kang G. Shin, *Real-Time Systems*, McGraw-Hill, 1997.
- Some real-time related conference and journal papers.

Evaluation:

- Theoretical Exercises: 2 points
- Mid-term and Final Exams: 15 points
- Practical Project: 3 points

4 General Mathematics 1 and 2

Course Code: 22015/22016, Units: 4/4, Prerequisites: None/General Mathematics 1

Course Objectives

1. Introduce students to differential and integral calculus as tools for solving problems, especially nonlinear ones.
2. Present concepts of n-dimensional linear algebra as a foundation for analyzing problems with n parameters.
3. Help students understand the principle of approximation and motivate computational solutions using calculators and computers.
4. Emphasize concepts and intuition while avoiding reliance on computational techniques that are easily performed by calculators or computers.
5. Avoid abstract concepts that lack motivation while focusing on providing a conceptual framework and essential tools for problem formulation and resolution.
6. Introduce the concept of differential equations and systems of differential equations naturally throughout the course, with applications such as growth, decay, oscillatory motions, and linear/nonlinear transformations.
7. Structure the topics based on educational goals to ensure content feels distinct from high school material and remains engaging.
8. Include topics that serve as prerequisites for Differential Equations and Engineering Mathematics, reducing the content load of those future courses.
9. Present the curriculum as a single-year course to allow flexibility in teaching both courses across academic years.

Course Topics

1. **Numbers:** Historical overview of numbers, rational and irrational numbers, completeness axiom, complex numbers, and some applications; sequences and series of numbers.
2. **Single-Variable Functions:** Limits and continuity; properties of continuous functions on closed intervals; differentiability, linear approximation, applications of derivatives, Taylor polynomials, and their applications.

3. **Single-Variable Integration:** Definite and indefinite integrals, fundamental theorems, transcendental functions, differential equations, approximation methods, traditional applications of integration, including an introduction to probability.

4. **Differential Equations:** Growth and decay problems, oscillatory motions.

5. **Functional Series:** Power series, Taylor series, Fourier series, and their applications, including solving differential equations using power series.

6. **Introduction to n-Dimensional Linear Algebra:** Properties of linear spaces ($\mathbb{R}^n$), inner product and applications, subspaces, linear mappings and applications, concepts of volume and determinants, diagonalization of symmetric matrices.

7. **Curves in Plane and Space:** Concepts of curvature and torsion; fundamental theorems.

8. **Functions from $\mathbb{R}^n$ to $\mathbb{R}^n$:** General properties, representations of multivariable functions, concepts of limits, continuity, and partial derivatives.

9. **Multivariable Differentiation:** Differentiability, gradients, chain rule, higher-order derivatives, multivariable Taylor polynomials and series, inverse function theorem, and implicit function theorem.

10. **Optimization:** Critical and regular points, classification of critical points, finding maxima and minima with and without constraints, Lagrange multipliers.

11. **Multiple Integration:** Basic concepts, computation, improper integrals, general variable transformation formula.

12. **Line Integrals and Vector Fields:** Basic concepts and applications, computation, conservative fields, and potentials.

13. **Surface Integrals on Curved Surfaces:** Analysis of smooth parametric and general surfaces, computation of surface integrals, and applications.

14. **Vector Analysis:** Concepts of divergence and curl with geometric and physical interpretations, Green's theorem, Stokes' theorem, and divergence theorem in various forms, with applications in scalar and vector potential problems.

5 Differential Equations

Course Code: 22034, Units: 3, Prerequisite: General Mathematics 2 (or concurrent enrollment)

Course Objectives

1. Emphasize modeling and studying mathematical models of physical, natural, and social systems.
2. Study differential equations using analytical, geometric, and qualitative methods.
3. Focus on conceptual understanding and intuition while avoiding reliance on computational techniques that can easily be performed using calculators or computers.
4. Use mathematical software to solve differential equations.

Course Topics

1. Solving ordinary differential equations (ODEs) using analytical, geometric, and qualitative methods.
2. Linear ODEs, especially second-order equations, linear independence of solutions, method of undetermined coefficients, and variation of parameters.
3. Systems of linear equations and the method of undetermined coefficients.
4. Nonlinear autonomous equations, singular points, stability, and asymptotic stability.
5. Lyapunov's second method.
6. Predator-prey problems.
7. Fourier series.
8. Partial differential equations (PDEs) of second order: heat equation, wave equation, and Laplace's equation.

6 Computer Architecture Lab

Course Code: 40103, Units: 1, Prerequisite: Computer Architecture, Logic Circuits Lab

Course Objectives

The primary objective of this course is to familiarize students with practical methods for implementing key components of a computer architecture (such as the arithmetic-logic unit, control unit, and memory). The course aims to provide students with a practical understanding of designing and implementing an instruction set architecture on a target architecture.

Course Topics

1. Introduction to CAD tools for designing and testing the correctness of logic circuits.
2. Introduction to a sample simulator (e.g., Quartus).
3. Design, implementation, and testing of a sample circuit (e.g., a circuit for adding two two-digit decimal numbers) using a simulator.
4. Design and implementation of computational architectures.
5. Design and implementation of a 4-bit fixed-point multiplier.
6. Design and implementation of a floating-point adder/subtractor.
7. Design and implementation of a decimal-to-binary converter.
8. Design and implementation of a simple computer architecture.
9. Design and implementation of a calculation unit with selectable source.
10. Design and implementation of a calculation unit with program-controlled operations.
11. Complete design and implementation of a computer with data memory and jump instructions.
12. Design and implementation of a processor.
13. Design and implementation of a micro-programmed control circuit.
14. Performance testing of the implemented circuit.

References

1. D. Patterson and J. L. Hennessy. *Computer Organization & Design, The Hardware/Software Interface*, 4th Edition, Morgan Kaufmann Publishing, 2011.

2. M. Mano. *Computer System Architecture*, 3rd Edition, Prentice Hall, 1992.

7 Discrete Structures

Course Code: 40115, Units: 3, Prerequisite: None

Course Objectives

This course aims to familiarize students with the concepts, structures, and techniques of discrete mathematics widely used in computer science and engineering. The goals include developing foundational skills such as understanding and constructing precise mathematical proofs, creative problem-solving, understanding basic results in logic, combinatorics, number theory, graph theory, and computation theory, and providing the mathematical prerequisites necessary for many other courses in various computer engineering disciplines.

Course Topics

1. **Logic (3 sessions)**
    - Basics of logic, propositions, equivalent propositions
    - Predicates, quantifiers, inference rules
    - Methods of proof
2. **Set Theory and Functions (2 sessions)**
    - Fundamentals of set theory, set operators, countable and uncountable sets
    - One-to-one and onto functions, function composition, inverse functions, sequences
3. **Number Theory (2 sessions)**
    - Divisibility, congruence, modular arithmetic
    - Prime numbers, Euler's theorem, introduction to cryptography
4. **Induction (2 sessions)**
    - Mathematical induction, well-ordering principle
    - Strong induction, structural induction
5. **Counting (4 sessions)**
    - Basic counting principles, permutations, and combinations
    - Binomial coefficients, permutations, and combinations with repetition

- Inclusion-exclusion principle, distributing objects into boxes
- Pigeonhole principle

6. **Discrete Probability (2 sessions)**
   - Probability theory, probability distribution functions, conditional probabilities
   - Random variables, expected value, variance
7. **Recurrence Relations (3 sessions)**
   - Recurrence problems
   - Solving recurrence relations (homogeneous and non-homogeneous)
   - Generating functions
8. **Relations (2 sessions)**
   - Relations and their properties, representation of relations, composition of relations
   - Equivalence relations, closures
9. **Partial Orders and Boolean Algebra (2 sessions)**
   - Partially ordered sets, Hasse diagrams, topological sorting
   - Lattices, Boolean algebra, properties of Boolean algebra
10. **Graphs (3 sessions)**
    - Basic definitions, special graphs, bipartite graphs, graph representation, graph isomorphism
    - Paths and connectivity, Eulerian and Hamiltonian paths
    - Planar graphs, Euler's formula, graph coloring
11. **Trees (1 session)**
    - Trees and forests, special trees, rooted trees, spanning trees
12. **Algebraic Structures (1 session, optional)**
    - Monoids, rings, groups, Abelian groups
13. **Modeling Computation (3 sessions)**
    - Languages and grammars, finite state machines
    - Language recognition, regular languages
    - (Optional) Turing machines

Evaluation

- Theory Assignments: 15%
- Exams (Midterm, Final, and Quizzes): 85%

References

1. K. H. Rosen. *Discrete Mathematics and Its Applications*. 8th Edition, McGraw Hill, 2018.
2. R. P. Grimaldi. *Discrete and Combinatorial Mathematics: An Applied Introduction*. 5th Edition, Pearson Addison Wesley, 2004.
3. A. Engel. *Problem-Solving Strategies*. Springer, 1998.

8 Fundamentals of Electrical and Electronic Circuits

Course Code: 40124, Units: 3, Prerequisite: Physics II

Course Objectives

This course aims to introduce students to electrical components and methods for analyzing electrical circuits in both the time and Laplace domains. It also provides an introduction to electronic circuits that constitute logic gates in several widely used technologies.

Course Topics

1. **Introduction to Electrical Circuits, Basic Elements, and Their Relations (8 sessions)**
    - Kirchhoff's voltage and current laws
    - Series and parallel connections of resistive elements
    - Circuit analysis methods: Node and mesh analysis
    - Linearity and the superposition principle
    - Thevenin and Norton equivalent circuits
    - Operational amplifiers and practical applications
2. **Circuit Analysis in the Time Domain (5 sessions)**
    - Introduction to waveforms (step, pulse, impulse, sinusoidal)
    - Introduction to energy storage elements and active elements
    - First-order electrical circuits
    - Second-order electrical circuits
3. **Circuit Analysis in the Frequency Domain (5 sessions)**
    - Laplace transform
    - Impedance and admittance
    - Circuit analysis using the Laplace transform
4. **Diodes and Transistors (2 sessions)**
    - Diode characteristics, models, and applications

- Logic inverter
- General characteristics and models of transistors

5. **Field-Effect Transistors (6 sessions)**
    - Structure, operation, and characteristics of enhancement-mode MOSFETs
    - Types of inverters using transistors
    - Pass transistors and transmission gates
    - Static CMOS logic
6. **Practical Circuits (4 sessions)**
    - Latches and flip-flops in static CMOS logic
    - Shift registers
    - Types of RAM and ROM memory
    - Analog-to-digital converters (ADC)
    - Digital-to-analog converters (DAC)

Evaluation

- Theory Assignments: 3 points
- Midterm and Final Exams: 14 points
- Quizzes: 3 points

References

1. William H. Hayt & Jack E. Kemmerly. *Engineering Circuit Analysis*. 7th Edition, McGraw-Hill, 2007.
2. Ernest Kuh & Charles Desoer, *Basic Circuit Theory*, Translated by Dr. Jabedar Maralani, University of Tehran Publications, 2016 (1395).
3. Adel Sedra & Kenneth Smith, *Microelectronic Circuits*, Translated by Majid Malekan & Haleh Vahedi, 4th Edition, Daneshgahi Sciences Press, 2002 (1381).
4. Mahmoud Tabandeh, *Pulse Techniques and Digital Circuits*, Sharif University Scientific Publishing Institute, 1997 (1376).

9 Computer Structure and Machine Language

Course Code: 40126, Units: 3, Prerequisites: Foundations of Programming, Logic Circuits

Course Objectives

The primary objective of this course is to familiarize students with the components of a computer and their interactions during the execution of program instructions. Programming in machine and assembly languages and their translation enable students to gain a deeper understanding of instruction set architectures and efficient utilization of machines. By the end of the course, students will be prepared to learn how to design and implement these components in the Computer Architecture course.

Course Topics

1. **Computer History**
    - Introduction to computer generations and types
    - Von Neumann model
2. **Data Representation**
    - Numbers: integer/fractional, signed/unsigned, fixed-point/floating-point, binary/decimal, etc.
    - Characters: 7-bit and 8-bit base codes, 16-bit and 32-bit universal codes
3. **Computer Structure**
    - Central Processing Unit (CPU), Arithmetic Logic Unit (ALU), Registers, Control Unit (CU), Main Memory
    - Shared Bus, Fetch-Execute Cycle
    - Addressing Modes: Immediate, Direct, Indirect, Relative, Implicit, Indexed, Segmented, Paged
4. **Assembly Language Programming and Translation to Machine Language on Simple Computers**
    - Assembler, Debugger, Compiler, Linker, Loader
    - Overview of the Instruction Set of at least one CISC computer (e.g., Intel 8086, IBM 360/370, or MC68000 recommended)

- Introduction to the computer structure and addressing methods
- Instruction set and assembly programming for the selected computer
- Structured programming constructs (subroutines, macros, etc.)
- Interrupts and their management

5. **RISC Computer Overview**
    - Overview of the Instruction Set of at least one RISC computer (e.g., MIPS recommended)
    - Introduction to the computer structure and addressing methods
    - Instruction set and assembly programming for the selected computer
    - Structured programming constructs (subroutines, macros, etc.)
    - Interrupts and their management

Evaluation

- Theory Assignments: 3 points
- Midterm and Final Exams: 15 points
- Quizzes: 2 points

References

1. J. L. Antonakos. *The 68000 Microprocessor: Hardware and Software Principles and Applications*. Prentice Hall, 2004.
2. M. A. Mazidi et al. *The x86 PC: Assembly Language, Design, and Interfacing*. Prentice Hall, 2010.
3. G. Struble. *Assembler Language Programming: The IBM System/360*. Addison-Wesley, 1971.
4. D. A. Patterson and J. L. Hennessy. *Computer Organization and Design MIPS Edition: The Hardware/Software Interface*. 5th Edition, Elsevier (Morgan Kaufmann), 2013.

10 Engineering Probability and Statistics

Course Code: 40181, Units: 3, Prerequisite: Calculus I

Course Objectives

This course aims to introduce students to fundamental concepts of probability theory and statistical inference and their applications in computer engineering, such as data modeling problems like regression. These concepts include interpretations and axioms of probability, single and multivariable probability distributions, conditional probability and statistical independence, random variables and expectations, functions defined on random variables, the exponential family of distributions, the Central Limit Theorem, the Law of Large Numbers, and hypothesis testing.

Course Topics

1. **Interpretations of Probability**
    - Axioms of Probability
    - Operations on Events
    - Statistical Independence, Conditional Probability, and Bayes' Rule
2. **Random Variables**
    - Expectation and Its Properties
    - Higher-Order Moments and Characteristic Functions
    - Functions of a Single Random Variable
3. **Joint Probability Distributions**
    - Joint Probability Density Function and Bayes' Law
    - Conjugate Prior Distributions
    - Exponential Family of Distributions
    - Special Distributions
    - Joint Moments
    - Functions of Two or More Random Variables
4. **Key Theorems**
    - Central Limit Theorem
    - Law of Large Numbers

- Convergence in Probability

5. **Estimation and Inference**
    - Maximum Likelihood and Maximum A Posteriori Probability Estimators
    - Bayesian Estimation
    - Properties of Estimators
6. **Statistical and Hypothesis Testing**

Evaluation

- Theory Assignments: 4 points
- Midterm Exam 1: 4 points
- Midterm Exam 2: 4 points
- Final Exam: 6 points
- Quizzes: 2 points

References

1. A. Poppulis and S. Pillai. *Probability, Random Variables and Stochastic Processes*. 4th Edition, McGraw Hill, 2002 (Chapters 1 through 8).
2. S. Ross. *A First Course in Probability*. 10th Edition, Prentice Hall, 2019.
3. G. Casella and R. L. Berger. *Statistical Inference*. 2nd Edition, Wadsworth Press, 2002.

11 Digital Systems Design Lab

Course Code: 40203, Units: 1, Prerequisite: Logic Circuits Lab, Corequisite: Digital Systems Design

Course Objectives

The objective of this laboratory is to provide students with hands-on experience in digital systems design using automatic digital design tools (CAD Tools) and the implementation of digital systems using programmable elements such as CPLD and FPGA.

Lab Topics

1. **Experiment 1:** Designing Combinational Circuits Using Schematic Tools
2. **Experiment 2:** Designing Sequential Circuits Using Schematic Tools
3. **Experiment 3:** Data Flow Description
4. **Experiment 4:** Behavioral Description
5. **Experiment 5:** Multiplier Design
6. **Experiment 6:** Incubator Design
7. **Experiment 7:** UART Design
8. **Experiment 8:** Complex Number ALU Design
9. **Experiment 9:** Implementation of Tri-state Buffers
10. **Experiment 10:** Simple Processor Implementation

References

1. S. Palnitkar. *Verilog HDL: A Guide to Digital Design and Synthesis.* 2nd Edition, Prentice Hall, 2003.
2. ACEX 1K Programmable Logic Family Data Sheet. Available at www.altera.com.
3. ModelSim User's Manual. Available at www.actel.com.
4. Introduction to the Quartus II Software. Available at www.altera.com.

12 Logic Design Lab

Course Code: 40206, Units: 1, Prerequisite: Logic Circuits, Corequisite: None

Course Objectives

The objective of this lab is to familiarize students with the implementation of logic circuits, including shift registers, adders, subtractors, counters, latches, and data buses. The Logic Design Lab provides practical experience with the theories learned in the Logic Circuits course.

Lab Topics

1. Familiarization with laboratory equipment and guides.
2. Understanding transfer characteristics and fan-out in TTL chips.
3. Familiarization with sequential circuits.
4. Timing circuits.
5. Shift registers.
6. Counters.
7. Design and implementation of finite state machines.
8. Implementation of a timer for a washing machine.
9. Implementation of a telephone system.
10. Familiarization with components of a simple computer.
11. Understanding the arithmetic and logic unit (ALU), registers, and buses.
12. Implementation of a hardware stack.
13. Design of a simple binary computer.

References

1. D. Patterson and J. L. Hennessy. *Computer Organization & Design, The Hardware/Software Interface.* 4th Edition, Morgan Kaufmann Publishing, 2011.
2. M. Mano. *Computer System Architecture.* 3rd Edition, Prentice Hall, 1992.

13 Logic Design

Course Code: 40212, Units: 3, Prerequisite: None, Corequisite: None

Course Objectives

The aim of this course is to introduce students to logic gates as circuit realizations of logical operators and simple integrated circuits constructed with a limited number of gates. In this course, students will learn methods for analyzing and designing combinational and synchronous sequential circuits. They will also become familiar with the structure, operation, and application of some simple integrated circuits, which form the building blocks of more complex systems, providing a foundation for understanding hardware components such as processors.

Course Topics

1. **Representation of Numbers in Base 2** (2 sessions)
    - Number base conversion
    - Representation of negative numbers using sign magnitude, 2's complement, and 1's complement
    - Addition and subtraction of numbers
    - Carry bits
    - BCD number representation
2. **Combinational Circuits** (3 sessions)
    - Boolean algebra and key algebra
    - Logic gates
    - Minterms and Maxterms
    - SOP and POS forms
    - Delay and critical path
3. **Simplification of Logical Functions** (4 sessions)
    - Algebraic methods
    - Karnaugh maps
    - Concepts of don't care and forbidden inputs
    - Implementation of two-level circuits

- Race, Hazard, and Glitch concepts
- Hazard elimination

4. **Combinational Components** (5 sessions)
    - Decoder and multiplexer
    - Implementing functions with decoders and multiplexers
    - Encoder and priority encoder
    - Demultiplexer
    - Half adder and full adder
    - Ripple-carry adder and carry-lookahead adder
    - Comparator
    - Read-Only Memory (ROM)
5. **Multivalue Logic** (2 sessions)
    - Three-state and four-state logic
    - Tri-state gates
    - Open-collector gates
    - Wired logic
    - Pull-up and pull-down resistors
6. **Sequential Circuits** (4 sessions)
    - Concept of sequential circuits
    - Types of Latches and forbidden inputs in Latches
    - Level-sensitive flip-flops, edge-sensitive flip-flops, and Master-Slave flip-flops
    - Synchronous and asynchronous reset inputs
    - Setup-time and Hold-time
7. **Finite State Machines (FSM)** (4 sessions)
    - Mathematical concept of FSM
    - State diagrams
    - State tables
    - Excitation tables

- Steps for implementing FSM
- Mealy and Moore models and their differences

8. **Sequential Components** (4 sessions)
   - Registers and shift registers
   - Universal registers
   - Synchronous counters
   - Johnson counters
   - Asynchronous counters (Ripple counters)
9. **Programmable Logic Devices (PLDs)** (2 sessions)
   - Introduction to PLDs
   - Introduction to SPLDs
   - PAL and PLA and comparison with ROM
   - PAL with tri-state logic

References

1. M. Morris Mano. *Digital Design*. 5th Edition, Prentice Hall, 2006.
2. Victor P. Nelson, H. Troy Nagle, Bill D. Carroll, David Irwin. *Digital Logic Circuit Analysis and Design*. Prentice Hall, 1995.
3. Franklin P. Prosser and David E. Winkel. *The Art of Digital Design: An Introduction to Top-Down Design*. Prentice Hall, 1987.
4. Alireza Ejlali. *Digital Circuits*, 1st Edition, Nasir Publishing, 1397 (Persian).

14 Numerical Computations

Course Code: 40215, Units: 3, Prerequisite: Differential Equations, Corequisite: None

Course Objectives

This course aims to familiarize undergraduate students with numerical methods for solving problems in various fields of science and engineering. These methods can approximate solutions to scientific problems where exact calculations are either impossible with conventional mathematical methods or computationally impractical due to complexity. In some cases, exact solutions can be computed but are highly complex, introducing errors. Numerical methods provide approximate solutions with limited error and reduced complexity. At the start of this course, students will be introduced to the concepts of error, and later, they will learn different numerical methods for solving engineering problems. Additionally, using efficient software environments to solve problems, comparing numerical methods, and graphically presenting results to summarize and conclude are also key objectives. The course will also include real-world problems whose solutions are challenging with traditional mathematical methods but are solvable using numerical methods.

Course Topics

1. **Introduction to Software Tools** (2 sessions)
    - MATLAB or Python usage
    - Matrix calculations
    - Vectors and plotting
    - Files and function definitions
    - Built-in functions in the chosen tool
2. **Errors** (4 sessions)
    - Introduction to errors
    - Floating-point representation
    - Sources of errors
    - Relative and absolute errors
    - Rounding, intrinsic, and truncation errors
    - Error propagation and graphical process of error propagation

- Numerical instability

3. **Solving Nonlinear Equations** (4 sessions)
    - Introduction to finding roots of nonlinear single-variable functions
    - Bisection method
    - Fixed-point iteration method
    - Secant method
    - Newton-Raphson method
    - Simple iteration or fixed-point method
    - Convergence rates of different methods
    - Necessary/sufficient conditions for convergence in Newton-Raphson, secant, and simple iteration methods
    - Horner's method for polynomial evaluation
    - Generalized Newton-Raphson method for solving systems of nonlinear equations
    - Intuitive and mathematical proofs for the above methods
4. **Interpolation, Extrapolation, and Curve Fitting** (5 sessions)
    - Introduction to interpolation, extrapolation, and curve fitting
    - Various interpolation methods including Lagrange's method, Newton's divided difference method, forward, backward, and central differences
    - Proof and error analysis for these methods
    - Polynomial curve fitting using the least squares method
    - Curve fitting using linearization techniques
    - Extrapolation
5. **Numerical Integration and Differentiation** (4 sessions)
    - Introduction to numerical integration and differentiation
    - Various methods of numerical integration including rectangle method, midpoint rule, trapezoidal rule, Gaussian quadrature, Simpson's 1/3 and 3/8 rules, and Romberg's method
    - Error analysis of the above methods
    - Mathematical and intuitive proofs for the methods

- Numerical differentiation using methods such as the midpoint rule, central difference, and three-point methods
- Error order analysis for the methods and using Richardson extrapolation to improve results in numerical differentiation

6. **Solving Initial Value Problems for Ordinary Differential Equations (ODEs)** (4 sessions)
    - Introduction to ODEs
    - Single-step methods including Taylor series method, Euler's method, modified Euler's method, second-order Runge-Kutta methods (Heun, midpoint, and modified Euler), third-order and fourth-order Runge-Kutta methods
    - Multistep methods like Adams-Moulton method
    - Error analysis and comparison of the above methods
    - Converting higher-order differential equations into systems of linear differential equations
    - Converting single-step methods for solving linear ODEs into numerical methods suitable for solving systems of linear ODEs
7. **Numerical Solution of Linear Systems of Equations** (4 sessions)
    - Introduction to solving systems of linear equations
    - Introduction to matrices
    - Direct methods for solving linear systems including matrix inversion, Cramer's rule, Gaussian elimination (forward, backward, and Gauss-Jordan elimination), and LU decomposition (Cholesky, Doolittle, and Crout decompositions)
    - Iterative methods such as Jacobi method and Gauss-Seidel method
    - Eigenvalues and eigenvectors, power method for estimating the dominant eigenvalue and corresponding eigenvector, and Gershgorin circle theorem

Evaluation

- **Exams**: Midterm and final exams (50% of total grade)
- **Assignments**: 6 theoretical assignments (solving scientific problems using numerical methods), 6 practical assignments using the introduced software tool (40% of total grade)

- **Project**: Project topic selected with the help of the instructor. The project can be research-oriented or practical (using the introduced tool). The research project must be reported in a report. Practical projects are preferred for gaining proficiency with the introduced tool. Group projects, if defined clearly, can have a significant positive impact on students' teamwork abilities (10% of total grade)

References

1. S. Pal. *Numerical Methods Principles, Analysis and Algorithms*. Oxford University Press, 2010.

2. J. Kiusalaas. *Numerical Methods in Engineering with Python 3*. Cambridge University Press, 2013.

3. J. Kiusalaas. *Numerical Methods in Engineering with MATLAB*. Cambridge University Press, 2015.

4. C. B. Moler. *Numerical Computing with MATLAB*. MathWorks, 2013.

15 Digital Systems Design

Course Code: 40223, Units: 3, Prerequisite: Computer Organization and Language, Corequisite: None

Course Objectives

This course aims to introduce students to the Verilog hardware description language (HDL), teaching them how to design hardware using HDLs at various levels of abstraction. It also focuses on the internal structure of programmable circuits and how hardware circuits can be implemented in FPGAs and CPLDs.

Course Topics

1. **ASM and FSM**
    - FSM diagrams and their applications
    - Modeling and synthesizing control circuits using FSM
    - ASM diagrams and designing digital systems using them
    - Control units and data paths
    - Synthesizing data paths from ASM diagrams
    - Methods of synthesizing control units from ASM diagrams
2. **Introduction to Hardware Description Languages (HDL)**
    - Overview of HDL languages
    - Key features and differences between concurrent and sequential HDL codes
    - The importance of using HDL languages
3. **Verilog Language Introduction**
    - Overview of Verilog language features
    - Comparison between Verilog and other HDLs
    - Reasons for using HDL languages in digital design
    - Basics of Verilog syntax and structure
    - Top-down and bottom-up design approaches
    - Structural and behavioral modeling
    - Introduction to modules, initial, always, and @ in Verilog

- Modular design and Verilog features for modularity

4. **Data Types and Basic Concepts in Verilog**
    - Wire and register types and their differences
    - Four-value logic and signal power concepts
    - Array and vector concepts and their applications
    - Real, Integer, Time, and String data types
    - Parameter and parameterized design in Verilog
    - System tasks, directives, and macros in Verilog
    - Hierarchical naming and its applications
5. **Structural Modeling in Verilog**
    - Ports in modules and types of ports
    - Mapping ports and connection rules
    - Gate-level design in Verilog
    - Delay modeling in structural description
    - Dataflow modeling in Verilog
    - Inertial vs. transport delay modeling
6. **Behavioral Modeling in Verilog**
    - Blocking and non-blocking assignments in behavioral modeling
    - Event control in behavioral modeling
    - Control structures such as decision-making and loops
    - Functions and tasks in Verilog
    - Types of event control (level-sensitive, edge-sensitive, named)
    - Timing control techniques
    - Modeling inertial and transport delays in behavioral modeling
    - Race conditions in concurrent statements
7. **Verilog Code Synthesis and Simulation**
    - Writing synthesizable Verilog code
    - Rules for synthesizable code (avoiding delays, division operators, etc.)
    - Methods to avoid combinational loops

- Impact of loops in behavioral descriptions on synthesis
- Three-state logic and its impact on synthesis

8. **Digital System Design with PLDs**
    - Features of digital systems
    - Levels of abstraction and modeling techniques
    - Applications of programmable logic devices (PLDs)
    - Overview of SPLDs and CPLDs
    - Structure of SPLDs (PAL, PLA, ROM)
    - Structure of CPLDs
    - Technologies for SPLD and CPLD fabrication
    - Case studies of CPLDs
    - FPGAs and their structure
    - Types of FPGAs (LUT-based and MUX-based)
    - FPGA technologies (Anti-fuse and SRAM-based)
    - Methods for constructing programmable connections in FPGAs
    - Case studies of FPGA technologies
    - Latest FPGA products and FPSoCs

Evaluation

- **Theoretical Exercises**: 3 points
- **Midterm and Final Exams**: 15 points
- **Quizzes**: 2 points

References

1. Samir Palnitkar. *Verilog HDL: A Guide to Digital Design and Synthesis.* 2nd Edition, SunSoft Press, 2003.
2. S. Brown, J. Rose. *FPGA and CPLD Architectures: A Tutorial.* IEEE Design and Test of Computers, pp. 42-57, 1996.
3. Altera Data Sheets. Available at `www.altera.com`.
4. Xilinx Data Sheets. Available at `www.xilinx.com`.
5. Actel Data Sheets. Available at `www.actel.com`.

16 Advanced Programming

Course Code: 40244, Units: 3, Prerequisite: Fundamentals of Programming, Corequisite: None

Course Objectives

This course introduces object-oriented programming concepts using the Java programming language. The intrinsic features of Java, its programming capabilities, and the differences in approach between Java and other similar languages are emphasized. Topics such as concurrent programming and software quality are also addressed. It is assumed that students have prior knowledge of at least one programming language and are familiar with problem-solving techniques, such as searching, sorting, and mathematical operations. The focus of this course is on object-oriented concepts.

Course Topics

1. **Introduction to Java (1 session)**
    - History of Java
    - Features of the Java language
    - Writing the first Java program
2. **Basic Programming Concepts in Java (3 sessions)**
    - Variables, methods, conditionals, loops
    - Primitive data types
    - Strings and arrays
3. **Introduction to Object-Oriented Design and Programming (2 sessions)**
    - History and evolution of programming paradigms to object-oriented programming
    - Basic concepts of object-oriented programming
    - Object-oriented design mindset
    - Encapsulation, interfaces, classes, packages, and access modifiers
4. **Object-Oriented Programming in Java (6 sessions)**
    - Defining classes in Java

- Objects in memory and memory management
- Object initialization and destruction
- Garbage collector in Java
- Passing parameters in Java
- Constructors and the `this` keyword
- Static members and the package concept
- Introduction to UML class diagrams

5. **Inheritance (2 sessions)**
    - Concept of inheritance
    - Access modifiers: `protected`, `abstract`, `super`
    - Multiple inheritance in other languages
6. **Polymorphism (1 session)**
    - Polymorphism using inheritance
    - The concept of `virtual` in C++
    - `final` members in Java
7. **Interfaces (1 session)**
    - Using interfaces
    - Multiple inheritance using interfaces
8. **Software Testing (1 session)**
    - Concept of software quality assurance and its importance
    - Unit testing and writing unit tests using JUnit
    - Testing exceptions using JUnit
    - Introduction to mocking and a mocking library in Java
9. **Design Patterns (1 session)**
    - Definition and significance of design patterns in software engineering
    - GoF design patterns and their categories
    - Detailed explanation of Singleton, State, Strategy, Observer, and Facade patterns

- Introduction to MVC architectural pattern

10. **Refactoring (2 sessions)**
    - What is refactoring and its importance for clean code
    - Signs of bad code
    - Refactoring patterns
    - Creating methods, transferring features between objects, organizing data
    - Simplifying conditional expressions, method calls, and error management
11. **Error and Exception Handling (2 sessions)**
    - Traditional error management model
    - Java's error management framework
    - Benefits of this model
    - `finally`, runtime exceptions
12. **Generics (1 session)**
    - Generic methods and classes
    - Applications of generics
    - Creating and using generic classes
    - Generics and inheritance
    - Difference between Java generics and C++ templates
13. **Collections and Containers (2 sessions)**
    - Data structures available in Java
    - Collections, ArrayList, LinkedList, Set, Map
    - Using Iterator
14. **File, Streams, and Networking (2 sessions)**
    - File I/O in Java
    - Serialization
    - Socket programming

15. **Concurrency (1 session)**

    - The need for concurrency
    - Concurrency in Java
    - Life cycle of a thread
    - Introduction to synchronization and critical sections

16. **Reflection (1 session)**

    - Need for runtime type information (RTTI)
    - RTTI in Java and its applications

17. **Advanced Concepts (1 session)**

    - Inner classes and anonymous classes
    - Annotations
    - Enumeration

Evaluation

- **Exams (Midterm, Final, and Quizzes)**: 50%
- **Programming Assignments**: 25%
- **Project (3 phases throughout the term)**: 25%

References

1. P. Deitel, H. Deitel. *Java: How to Program.* 11th Edition, Pearson Education, 2017.
2. B. Eckel. *Thinking in Java.* 4th Edition, Prentice Hall, 2006.
3. M. Fowler, K. Beck, J. Brant, W. Opdyke, D. Roberts. *Refactoring: Improving the Design of Existing Code.* Addison-Wesley, 1999.

17 Data Structures and Algorithms

Course Code: 40254, Units: 3, Prerequisite: Discrete Structures, Corequisite: Advanced Programming

Course Objectives

In this course, students will become familiar with methods of algorithm analysis, simple and moderately advanced but important data structures, and some basic algorithms. Emphasis will be placed on analyzing and proving the correctness of algorithms. The course assumes prior knowledge of programming in either C++ or Java and familiarity with recursive problem-solving techniques. The algorithms are taught independently of the language and according to the reference textbook.

Course Topics

1. **Introduction (1 session)**
    - Levels of abstraction
    - Problem-solving steps and abstraction
    - Data models, data types, data structures, abstract data types, objects
2. **Algorithm Analysis (3 sessions)**
    - Time complexity analysis: insertion sort
    - Growth of functions
    - Divide-and-conquer analysis
3. **Divide and Conquer (2 sessions)**
    - Merge sort, counting inversions, longest common subsequence, matrix multiplication
    - Master theorem for divide and conquer recurrences
4. **Randomized Algorithms (1 session)**
    - Approximate median calculation, hiring problem
5. **Basic Data Structures (1 session)**
    - Queue and stack
    - Linked list

6. **Tree Data Structures (5 sessions)**
    - Different implementations of trees, tree traversal, structural induction
    - Expression trees, converting between different notations of a mathematical expression
    - Trie data structure
    - Binary search tree
    - Priority queue (min-heap and max-heap)
7. **Sorting (4 sessions)**
    - Decision tree and lower bounds
    - Heap sort
    - Quick sort (randomized analysis)
    - Optimal number of comparisons in sorting
    - Linear time sorting: counting, radix, bucket sort
    - External sorting (optional)
8. **Order Statistics (2 sessions)**
    - Finding minimum and maximum
    - Selecting the k-th smallest element (randomized and deterministic algorithms)
9. **Hashing (2 sessions)**
    - Chaining in hashing
    - Open addressing in hashing
    - Perfect hashing
10. **Advanced Data Structures (3 sessions)**
    - Disjoint sets
    - Balanced binary trees: Red-Black trees
    - Interval tree
11. **Graphs (3 sessions)**
    - Different graph representations

- Depth-first search (DFS) and breadth-first search (BFS) and their applications
- Topological sort, strongly connected components
- Shortest path algorithms: Dijkstra's and Bellman-Ford algorithms

Evaluation

- **Five exercise sets**: Each set includes several theoretical problems and programming tasks. Theoretical problems do not need to be submitted.
- **Five short quizzes**: Based on theoretical problems above + a similar problem (3 points)
- **Five practical assignments**: (3 points)
- **Midterm exam**: (6 points)
- **Final exam**: (8 points)

References

1. Mohammad Qadsi, *Data Structures and Foundations of Algorithms*, 4th Edition, Fatemi Publications, 2014.
2. Mohammad Qadsi and Aidin Nasiri Shargh, *600 Multiple Choice Questions on Data Structures and Algorithms*, 6th Edition, Fatemi Publications, 2018.
3. T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd Edition, MIT Press, 2011.

18 Linear Algebra

Course Code: 40282, Units: 3, Prerequisite: Calculus II

Course Objectives

The goal of this course is to introduce students to the fundamental concepts of linear algebra and how to apply and implement them in an appropriate software environment. Familiarity with the concepts of this course enables the analysis of linear transformations and systems using matrices, operators, and related definitions. Optimization problems, as one of the most common applications of linear algebra, will also be explored.

Course Topics

1. **Vector Spaces**
    - Linear transformations and matrices
    - Vector space of linear transformations
    - Algebraic structure of linear transformations
2. **Matrices and Rank**
    - Inverse of linear transformations
    - Duality
3. **Linear Systems**
    - Volume and determinant
4. **Polynomials**
    - Zeros of polynomials
    - Factorization of polynomials in complex and real fields
5. **Eigenvalues and Eigenvectors**
    - Subspaces of invariants
    - Eigenvectors and eigenvalues
    - Linearly independent eigenvectors
6. **Inner Product Spaces**
    - Inner product and distance definition

- Orthogonal bases
- Operators in inner product spaces

7. **Operators and Decompositions**
    - Polar decomposition
    - Singular value decomposition (SVD)
    - Cholesky decomposition
    - LU decomposition
    - QR decomposition
    - Adjoints and normal operators
    - Isometric and isometric operators
    - Positive operators

Evaluation

- **Assignments**: 6 points
- **Two midterm exams**: 8 points
- **Final exam**: 6 points
- **Quizzes**: 1 point

References

1. Sheldon Axler, *Linear Algebra*, Springer, 2015.
2. Gilbert Strang, *Linear Algebra and Its Applications*, 4th Edition, Cengage Learning, 2006.
3. David Clay, *Linear Algebra and Its Applications*, 4th Edition, Pearson, 2011.

19 Computer Architecture

Course Code: 40323, Units: 3, Prerequisite: Computer Organization and Language

Course Objectives

In the Computer Organization and Language course, students are introduced to the various components of a computer and how they interact to execute the instructions of a program. The primary objective of this course is to teach how to design and implement these components, and the various techniques used in implementing different architectures for different applications.

Course Topics

1. **Overview of Basic Components and History of Computers**
    - Combinational and sequential circuits
    - Advantages of digital technology over analog
    - Multiplexers, decoders, tri-state gates, buses
2. **Abstraction Levels and Computer Description**
    - History and generations of computers
3. **Number Representation**
    - Different methods for digital representation of signed and unsigned numbers, integers and floating-point numbers
    - Absolute and relative precision, representation ranges
4. **Processor and Computer Performance**
    - Factors affecting computer performance
    - Definition of performance (inverse of execution time)
    - Performance formula
    - Benchmarking and examples
5. **Designing the Execution Unit (Data Path) and Hardwired Control**
    - Overview of addressing modes
    - Transfer levels and languages between registers (RTL)

- Instruction Set Architecture (ISA)
- Step-by-step analysis and design of a sample processor (MIPS)

6. **Interrupt Implementation and Polling Method**
   - Interrupt handling and methods
7. **Control Unit Description and Design**
   - Microprogrammed control unit
   - Comparison of advantages and disadvantages of microprogrammed control versus hardwired control
8. **Memory Systems**
   - Overview of memory types and hierarchy
   - Cache memory and different mapping methods (direct-mapped, fully associative, set-associative)
9. **Arithmetic Algorithms**
   - Algorithms for addition, subtraction, multiplication, and division
   - Booth's encoding and array multiplier for multiplication
10. **I/O Methods**
    - Handshaking techniques
11. **Advanced Architectures**
    - Overview of acceleration and parallelism methods
    - Pipeline architecture and execution time considerations

Evaluation

- **Theoretical Assignments**: 3 points
- **Midterm and Final Exams**: 15 points
- **Quizzes**: 2 points

References

1. D. A. Patterson and J. L. Hennessey, *Computer Organization and Design*, 3rd Edition, Elsevier (Morgan Kaufmann), 2005.
2. M. Mano, *Computer System Architecture*, 3rd Edition, Prentice Hall, 1992.

20 Design of Programming Languages

Course Code: 40364, Units: 3, Prerequisite: Advanced Programming

Course Objectives

The main objectives of this course are:

1. A review of the natural evolution of concepts and methods in the design and implementation of different generations of programming languages in an empirical and step-by-step approach.
2. Introduction to engineering methods for designing and implementing programming languages, particularly focusing on domain-specific languages (DSLs) and the importance of their design and implementation.
3. Familiarity with the implementation of interpreters, particularly in the context of virtual machines.
4. A review of the principles and issues related to programming language design, along with the data structures used in implementing or realizing a programming environment.

Course Topics

1. **Introduction**
    - Evolutionary history of programming languages and introduction of some significant languages from a historical perspective.
    - Comparative introduction of the major programming paradigms (imperative-procedural, object-oriented, rule-based programming, and declarative-functional programming) and their approach to the concept of a program.
2. **Interpretation vs. Compilation**
    - Comparing interpretation and compilation from both language design and implementation perspectives.
3. **Language Engineering**
    - Familiarity with existing tools for designing domain-specific languages and implementing efficient interpreters. Special focus: practical exercises with DrRacket.

4. **Functional Programming**

   - Review of core functional programming concepts and lambda calculus, along with practical exercises and projects. Suggested languages: Scheme (based on syntax and semantics from Friedman's book) or Racket (based on Krishnamurthi's book).
   - Modern interpretation of Lisp-based programming on programmable platforms.
   - Optional introduction to functional programming features in Java 8 and above.

5. **Designing and Implementing a Programming Language Interpreter**

   - Language with computational expressions (no side effects).
   - Adding non-recursive and recursive procedures (subprograms) to the language and its interpreter.
   - Adding concepts of variable scope and binding domains to the language and its interpreter.
   - Adding memory manipulation (reference-type variables) to the language and its interpreter.
   - Introducing type systems (type-annotated variables) in the language and its interpreter.
   - Implementing modular and object-oriented programming features (modules, classes, and objects) in the designed language and its interpreter.

6. **Selected Advanced Topics**

   - Overview of notable programming languages such as ML, Haskell, Scala, and F#.
   - Review of external factors influencing language design and implementation, such as requirements for parallel or concurrent programming, real-time systems, web-based programming, and component-based or service-oriented software engineering.
   - Introduction to programming language semantics and reasoning based on them:
     - Operational semantics
     - Denotational semantics
     - Axiomatic semantics (Hoare logic)

Evaluation

- **Midterm Exam**: 25%
- **Final Exam**: 40%
- **Assignments**: 20%
  - Functional programming exercises
  - Step-by-step interpreter design exercises
  - Theoretical assignments
- **Project**: 15%

References

1. D. P. Friedman, M. Wand, *Essentials of Programming Languages*, 3rd Edition, MIT Press, 2008.
2. S. Krishnamurthi, *Programming Languages: Application and Interpretation*, 2nd Edition, 2017.
3. M. Felleisen, R. B. Findler, M. Flatt, S. Krishnamurthi, E. Barzilay, J. McCarthy, S. Tobin-Hochstadt, *A Programmable Programming Language*, Communications of the ACM, Vol. 61, No. 3, pp. 62-71, March 2018.
4. Racket programming language and its toolkits.

21 Database Design

Course Code: 40384, Units: 3, Prerequisite: Data Structures and Algorithms

Course Objectives

In this course, students will become familiar with the concepts of data modeling and database design. By the end of the semester, students are expected to have a complete understanding of the topics outlined in the syllabus.

Course Topics

1. **Database Approach and Database Systems** (3 sessions)
    - Introduction to the course
    - Definition of databases
    - File-based approach vs. database approach
    - Components of a database environment
    - Types of database system architectures (centralized, client-server, distributed)
    - Components of relational DBMS (RDBMS, OLTP)
2. **Semantic Data Modeling using ER and EER** (4 sessions)
    - Entity
    - Attribute
    - Relationship
    - ER and EER diagrams
    - Types of constraints
    - Techniques such as allocation, generalization, decomposition, combination, and aggregation
    - Characteristics of semantic modeling techniques
3. **Principles of Database Design** (2 sessions)
    - Introduction to tabular structures and relational databases
    - Top-down design methodology (transforming semantic models into logical designs)

4. **Introduction to SQL and Database Implementation** (3 sessions)
    - Introduction to SQL language
    - DDL and DCL commands
    - DML commands
    - SQL integration in programming languages
    - Transaction implementation
    - Parameterized queries
5. **Three-Tier Database Architecture** (3 sessions)
    - ANSI three-tier architecture
    - View (perceptual) layer
    - Internal and external views
    - Transformations between layers
    - Types of indices (B-Tree, B+-Tree, and Hash)
    - Operations at the external level and related challenges
    - Physical and logical data independence
6. **Fundamental Concepts of Relational Data Model** (2 sessions)
    - Relational model components
    - Relation and related concepts
    - Keys in the relational model
    - Overview of relational database design principles
7. **Integrity Constraints in Relational Model** (2 sessions)
    - General integrity constraints (C1, C2)
    - User-defined integrity constraints
    - Mechanisms for applying user constraints (Assertions and Triggers)
8. **Operations in Relational Database** (3 sessions)
    - Relational algebra
    - Relational calculus
9. **Theory of Dependencies and Normalization** (3 sessions)

- Dependency theory concepts
- Normal forms (up to BCNF; other levels for individual study)
- Optimal decomposition

10. **Database Security** (1 session)
    - User management
    - Authentication
    - Access control
    - Data encryption
11. **NoSQL Database Systems** (2 sessions)
    - Reasons for using NoSQL databases
    - CAP theorem
    - Key-value, column-oriented, graph-oriented, and document-oriented NoSQL databases
12. **(Optional) Introduction to Data Warehousing** (1 session)
    - Introduction to Data Warehouses
    - OLAP concepts

Evaluation

- **Midterm Exam**: 30%
- **Final Exam**: 35%
- **Assignments**: 17%
- **Project**: 13%
- **Quizzes and Class Activities**: 5%

References

1. Mohammad Taghi Rouhani Rankouhi, *Fundamentals of Databases*, 4th Edition, 2011.
2. R. Elmasri, S. Navathe, *Fundamentals of Database Systems*, 8th Edition, Pearson, 2019.

3. A. Silberschatz, H. F. Korth, S. Sudarshan, *Database System Concepts*, 6th Edition, McGraw-Hill, 2010.

4. C. J. Date, *An Introduction to Database Systems*, 8th Edition, Pearson, 2003.

5. T. Connolly, C. Begg, *Database Systems*, 6th Edition, Pearson, 2014.

6. R. Ramakrishnan, J. Gehrke, *Database Management Systems*, 4th Edition, McGraw-Hill, 2014.

22 Operating Systems Lab

Course Code: 40408, Units: 1, Prerequisite: Operating Systems

Course Objectives

The objective of this lab is to teach the various components of the Linux operating system, how to utilize them, and implement algorithms for each of these components. After completing this lab, students will be familiar with the structure of the Linux operating system and will be capable of modifying and compiling it. The general topics of the lab are as follows, although the specifics of each experiment may vary from semester to semester. Basic experiments will always be covered, and then the focus will shift to various topics.

Course Topics

1. **Compiling and Installing Linux**
2. **Programming with C++ and Shell in Linux**
3. **Using Linux System Calls in Programs**
4. **Examining Operating System Behavior (proc/ directory)**
5. **Creating, Running, and Terminating Processes and Threads (using pthread library)**
6. **Process and Thread Synchronization and Communication**
7. **Memory Management, Shared Memory, and Virtual Memory**
8. **CPU Scheduling**
9. **Using Installable File Systems**
10. **Disk Scheduling and I/O Scheduling**
11. **Design and Implementation of Device Drivers**
12. **Using Linux Security Mechanisms**
13. **Introduction to Real-Time and Embedded Operating Systems**
14. **Introduction to Windows Research Kernel**

References

1. P. J. Salzman, M. Burian, and O. Pomerantz, *The Linux Kernel Module Programming Guide*, 2007.

2. K. Wall, M. Watson, and M. Whitis, *Linux Programming Unleashed*, Macmillan Computer Publishing, 1999.

3. M. Mitchell, J. Oldham, and A. Samuel, *Advanced Linux Programming*, New Rivers, 2001.

4. C. S. Rodriguez, G. Fischer, and S. Smolski, *The Linux® Kernel Primer: A Top-Down Approach for x86 and PowerPC Architectures*, Prentice-Hall, 2005.

5. J. Corbet, A. Rubini, and G. Kroah-Hartman, *Linux Device Drivers*, O'Reilly Books, 2005.

23 Compiler Design

Course Code: 40414, Units: 3, Prerequisite: Data Structures and Algorithms

Course Objectives

Compiler design and construction is one of the fundamental concepts in computer science. Although the methods for building compilers are somewhat limited, they can be used for building interpreters and translators for a wide variety of languages and machines. In this course, the subject of compiler construction is introduced through the description of the main components of a compiler, their duties, and interactions. After an introduction to the components of a compiler and types of grammars, various translation phases, such as lexical, syntactic, and semantic analysis, as well as code generation and optimization, are discussed.

Course Topics

1. **Introduction** (2 sessions)
2. **Types of Languages and Grammars** (1 session)
3. **Lexical Analysis and Error Handling** (3 sessions)
4. **Top-Down Syntactic Analysis** (5 sessions)
    - Recursive Descent Parsing
    - LL(1) Parsing
    - Error Handling in Syntax Analysis
5. **Bottom-Up Syntactic Analysis** (8 sessions)
    - Operator Precedence
    - Simple Precedence
    - LR(1) Parsing including SLR(1), LALR(1), and CLR(1)
6. **Semantic Analysis** (1 session)
7. **Symbol Table Management** (1 session)
8. **Run-Time Memory Allocation Techniques** (2 sessions)
9. **Code Generation** (5 sessions)

10. **Code Optimization and Refinement** (1 session)

11. **Automatic Compiler Generation** (1 session)

Evaluation

- Midterm Exam: 35%
- Final Exam: 35%
- Practical Project: 20%
- Quizzes and Exercises: 10%

References

1. A. Aho, M. Lam, R. Sethi, and J. Ullman, *Compilers: Principles, Techniques, and Tools*, 2nd Edition, Addison Wesley, 2007.

2. D. Grune, H. Bal, C. Jacobs, and K. Langendoen, *Modern Compiler Design*, John Wiley, 2001.

3. J. Tremblay and P. Sorenson, *Theory and Practice of Compiler Writing*, McGraw Hill, 1985.

4. C. Fisher and R. LeBlanc, *Crafting a Compiler with C*, Benjamin Cummings, 1991.

24 Theory of Machines and Languages

Course Code: 40415, Units: 3, Prerequisite: Data Structures and Algorithms

Course Objectives

This course covers theoretical aspects of computer engineering and computer science. The topics include various computational models, their computational power, computational properties, and their applications. Other topics include the concepts of computability, decidability, and the Church-Turing thesis regarding algorithms.

Course Topics

1. **Preliminary Topics** (4 sessions)

    - Propositional Logic, Predicate Logic, Proof Systems, Set Theory, Russell's Paradox, Countable and Uncountable Sets, Languages, and Grammars.

2. **Finite State Machines** (8 sessions)

    - Deterministic Finite Automata (DFA), Non-deterministic Finite Automata (NFA), Regular Languages, Regular Expressions, Right Linear Grammars, Left Linear Grammars, Regular Grammars, Context-free Grammars, Non-regular Languages, Pumping Lemma for Regular Languages.

3. **Context-Free Languages** (10 sessions)

    - Context-Free Grammars, Context-Free Languages, Leftmost Derivations, Rightmost Derivations, Derivation Trees, Ambiguous Grammars, Unambiguous Grammars, Intrinsically Ambiguous Languages, Non-ambiguous Languages, Simplification of Context-Free Grammars, Chomsky Normal Form, Greibach Normal Form, Membership Problem, CYK Algorithm, Pushdown Automata, Equivalence of Pushdown Automata and Context-Free Grammars, Deterministic Pushdown Automata, Deterministic Context-Free Languages, Non-context-free Languages, Pumping Lemma for Context-Free Languages.

4. **Computability** (8 sessions)

- Turing Machines, Church-Turing Thesis, Decidability and Undecidability, Computability and Uncomputability, Halting Problem, Post Correspondence Problem, Computational Complexity, Class P, Class NP, NP-Complete Problems, NP-Hard Problems.

Evaluation

- Weekly Exercises: 30%
- Quizzes: 45%
- Final Exam: 25%

References

1. M. Sipser, *Introduction to the Theory of Computation*, 3rd Edition, Cengage Learning, 2013.
2. P. Linz, *An Introduction to Formal Languages and Automata*, 3rd Edition, Jones and Bartlett Publishers, 2001.
3. J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, 2nd Edition, Addison-Wesley, 2001.
4. J. P. Denning, J. B. Dennis, and J. E. Qualitz, *Machines, Languages, and Computation*, Prentice-Hall, 1978.
5. J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, 1979.
6. P. J. Cameron, *Sets, Logics and Categories*, Springer, 1998.

25 Artificial Intelligence

Course Code: 40417, Units: 3, Prerequisite: Data Structures and Algorithms, Engineering Probability and Statistics

Course Objectives

This course introduces both theoretical and practical aspects of Artificial Intelligence (AI). The goal of the course is to introduce techniques for making optimal or near-optimal decisions in various problems and environments. The course covers topics such as search, problem-solving, knowledge representation, and inference. It also addresses search in uncertain environments, knowledge representation in these environments, and probabilistic inference for decision-making under uncertainty. Furthermore, an introduction to machine learning will be provided. The course will conclude with an introduction to several application areas of AI.

Course Topics

1. **Introduction to AI and its History**
2. **Introduction to Intelligent Agents**
3. **Uninformed Search**:
    - Breadth-First Search (BFS), Depth-First Search (DFS)
    - Iterative Deepening, Uniform Cost Search
4. **Informed Search**:
    - Admissible and Consistent Heuristics
    - Greedy Best-First Search
    - A* Algorithm and Optimality Proof
    - Heuristic Function Generation
5. **Local Search**:
    - Hill-Climbing, Simulated Annealing, Local Beam Search, Genetic Algorithms
    - Gradient Descent in Continuous Spaces
6. **Constraint Satisfaction Problems (CSP)**:

- Backtracking Search, Techniques like LCV, MRV, Forward Checking, MAC, AC3
- Local Search for CSP

7. **Adversarial Search**:
   - Minimax Algorithm and Alpha-Beta Pruning
   - Expectiminimax Algorithm
8. **Markov Decision Processes (MDP)**:
   - Policy Evaluation and Improvement
   - Value Iteration and Policy Iteration
9. **Reinforcement Learning**:
   - Model-Based Methods, Temporal Difference Learning, Q-Learning Algorithm
10. **Logic**:
    - Propositional Logic and Inference (including Resolution)
    - First-Order Logic and Inference
11. **Bayesian Networks**:
    - Representation in Bayesian Networks, Independence in Networks
    - Exact and Approximate Inference using Sampling
    - Parameter Estimation in Bayesian Networks
    - Applications: Markov Models, Hidden Markov Models, Naïve Bayes Classifier
12. **Introduction to Machine Learning**
    - Linear Models, Neural Networks
13. **Applications of AI**:
    - Natural Language Processing
    - Computer Vision
    - Robotics

Evaluation

- Theoretical and Practical Exercises: 6 points
- Midterm Exam: 5 points
- Final Exam: 7 points
- Quizzes: 2 points

References

- Stuart Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach*, 3rd Edition, 2009.

26 Computer Networks Lab

Course Code: 40416, Units: 1, Prerequisite: None, Corequisite: Computer Networks

Course Objectives

The Computer Networks Lab, offered for undergraduate students, serves as a complementary course to the Computer Networks lecture. In this course, students will gain practical experience with key concepts learned in the Computer Networks lecture. The lab is conducted in ten 3-hour sessions.

Course Topics

1. **Introduction to Basic Computer Network Concepts**
2. **Review of Layered Architecture**
3. **Physical Connection of Machines and Types of Transmission Cables**
4. **Network Cable Socketing**
5. **Introduction to Wireshark Software**
6. **Examination of HTTP Communication**
7. **Examination of TelNet Communication**
8. **Analysis of DNS Requests and Responses**
9. **Advanced Wireshark Usage**
10. **Configuring and Setting Up a DNS Server**
11. **Introduction to Routers and Switches**
12. **Introduction to Packet Tracer Software**
13. **Cisco Router and Switch Commands**
14. **Introduction to GNS3 Software**
15. **IP Addressing and IP Subnetting**
16. **Implementation of a Static Routing Scenario in Packet Tracer**

17. **Dynamic Routing**
18. **Configuring RIP Routing Protocol in Packet Tracer**
19. **Configuring OSPF Routing Protocol in Packet Tracer**
20. **Introduction to NAT Mechanism**
21. **Configuring Static NAT**
22. **Configuring Dynamic NAT**
23. **Configuring PAT**
24. **Introduction to BGP Routing Protocol**
25. **Implementing a BGP Protocol Scenario**

Evaluation

- Laboratory Activities and Reports: 15 points
  - 15 points will be awarded for completing the lab experiments during the sessions and submitting the corresponding reports before the next session.
  - Each session's score will be divided equally between the lab activity and the submission of the report.
  - Failure to attend a session results in losing the points for that session and report.
  - Before each session, students must read the experiment instructions and, if necessary, review related content from the Computer Networks lecture.
- Final Exam: 5 points

References

- James Kurose and Keith Ross, *Computer Networking: A Top-Down Approach*, 7th Edition, Pearson, 2016.
- Larry L. Peterson and Bruce S. Davie, *Computer Networks: A Systems Approach*, 5th Edition, 2011.
- Andrew Tanenbaum, *Computer Networks*, 5th Edition, Pearson, 2010.

27 Operating Systems

Course Code: 40424, Units: 3, Prerequisite: Computer Architecture

Course Objectives

The goal of this course is to familiarize undergraduate students with the principles of operating systems. The course includes four individual programming assignments that introduce students to system programming. Additionally, three group programming assignments will familiarize students with kernel-level programming.

Course Topics

1. **Introduction to Operating Systems** (2 sessions)
    - Basic concepts of operating systems
    - Structure and components of operating systems
    - Process, address space, I/O, and dual-mode operations
    - System architecture and structure
2. **Process Management** (3 sessions)
    - Single-threaded, multi-threaded processes, forked processes, and process control blocks
    - Interrupt management
    - Process communication
3. **Concurrency and Synchronization** (3 sessions)
    - Critical regions and mutual exclusion
    - Indivisible operations
    - Locks, semaphores, and monitors
4. **Scheduling** (3 sessions)
    - Scheduling algorithm objectives
    - First-Come-First-Served (FCFS), Round Robin (RR), Shortest Job Next (SJN), and Least Time Remaining First (LTRF)
    - Real-time scheduling

5. **Deadlock and Starvation** (2 sessions)
    - Conditions for deadlock occurrence
    - Methods for deadlock handling, detection, and prevention
    - Dining Philosophers problem and Banker's algorithm
6. **Memory Management** (2 sessions)
    - Memory management, paging, segmentation, and combination of paging and segmentation
    - Address translation, page table, two-level and multi-level paging, and inverted page tables
    - Translation Lookaside Buffer (TLB)
7. **Virtual Memory** (2 sessions)
    - Demand paging
    - Page frame allocation and page faults
    - Page replacement algorithms (FIFO, Least Recently Used, Random, Not Recently Used, Clock, and Nth Chance)
    - Working set and thrashing
8. **Mass Storage Systems** (2 sessions)
    - Types of I/O devices, controllers, and device drivers
    - Storage devices (HDD and SSD)
    - Disk scheduling algorithms (FCFS, Shortest Seek Time First, SCAN, and C-SCAN)
9. **File Systems** (3 sessions)
    - Disk management methods and components of file systems
    - File Allocation Table, UNIX file system, and NTFS
    - Memory-mapped files and caching in file systems
10. **Protection and Security** (1 session)
11. **Virtual Machines** (1 session)

Evaluation

- Midterm and final exams: 40% of the total grade
- Four individual programming assignments due throughout the semester: 25% of the total grade
- Three group programming assignments due throughout the semester: 35% of the total grade

References

- A. Silberschatz, P. B. Galvin, and G. Gagne, *Operating System Concepts*, 10th Edition, Wiley Publishing, 2018.
- T. Anderson and M. Dahlin, *Operating Systems: Principles and Practice*, 2nd Edition, Recursive Books, 2014.

28 Data and Network Security

Course Code: 40442, Units: 3, Prerequisite: Computer Networks (40443)

Course Objectives

The objective of this course is to familiarize students with the fundamental concepts of security, defense mechanisms, and attacks in the areas of system, web, network, and mobile security.

Course Topics

1. **Basic Concepts and Definitions**
    - Security policies and access control models
    - Covert channels, information flow control
    - Discretionary Access Control (DAC), Mandatory Access Control (MAC) models
    - Role-Based Access Control (RBAC)
2. **System Security**
    - Software execution methods and system interactions, vulnerabilities
    - Attacks and defensive methods (control hijacking)
    - Secure management of legacy code (sandboxing, virtualization, isolation at various layers)
    - Secure code development methods (static analysis, dynamic analysis)
    - Security violations and Fuzzing
3. **Web Security Model**
    - Security of web application software (SQL injection, XSS, CSRF)
    - Web session management (Cookies)
    - Symmetric and asymmetric cryptography concepts
    - Message integrity codes and hash functions
    - Web information security during transfer (Https/SSL)
    - Defensive mechanisms on the browser side (SOP, CSP, CORS)
4. **Network Security**

- Security threats in network protocols (routing, BGP, DNS, TCP, etc.)
- Defensive tools in networks (IDS, VPN, Firewalls)
- Denial of Service (DoS) attacks and defensive strategies
- Trusted Computing and SGX

5. **Mobile Security**
    - Security of mobile platforms (iOS, Android)
    - Mobile threats

Evaluation

- Theoretical exercises: 8% of the total grade
- Midterm and final exams: 10% of the total grade
- Quizzes: 2% of the total grade

References

- Matt Bishop, *Computer Security*, Addison-Wesley, 2017.
- John Erickson, *The Art of Exploitation*, 2nd Edition, No Starch Press, 2008.
- Robert C. Seacord, *Secure Coding in C and C++*, 2nd Edition, Pearson Education, 2005.
- A. Sotirov, *Bypassing Browser Memory Protections*, 2008.
- T. Garfinkel, *Traps and Pitfalls: Practical Problems in System Call Interposition Based Security Tools*, NDSS, 2003.
- Adam Barth, Collin Jackson, and John C. Mitchell, *Securing Browser Frame Communication*, Usenix, 2008.
- Adam Barth, Collin Jackson, Charles Reis, and the Google Chrome Team, *The Security Architecture of the Chromium Browser*, 2008.
- Bortz et al., *Origin Cookies: Session Integrity for Web Applications*, 2011.
- Enck, Ongtang, and McDaniel, *Understanding Android Security*, 2009.
- Allan Tomlinson, *Introduction to the TPM: Smart Cards, Tokens, Security and Applications*, 2008.

- Andrew Baumann, Marcus Peinado, and Galen Hunt, *Shielding Applications from an Untrusted Cloud with Haven*, OSDI 2014.

29 Computer Networks

Course Code: 40443, Units: 3, Prerequisite: Engineering Statistics and Probability, Co-requisite: Operating Systems

Course Objectives

The objective of this course is to introduce students to the fundamental concepts of computer networks and related topics.

Course Topics

1. **Socket Programming**
2. **IP Packet Switching**
3. **IP Addressing and Routing**
4. **Transmission Protocols (TCP and UDP)**
5. **Congestion Control**
6. **Address Translation (DNS, DHCP, and ARP)**
7. **Middleware**
8. **Switches and Bridges**
9. **Links**
10. **Connection-Oriented Routing**
11. **Distance Vector Routing and Path Vector Routing**
12. **Policy-Based Path Vector Routing (BGP)**
13. **Subnet Networks and Peer-to-Peer Networks**
14. **Multimedia Streaming**
15. **Circuit Switching**
16. **Wireless and Mobile Networks**
17. **Content Delivery Networks (CDN)**
18. **Software-Defined Networking (SDN)**

Evaluation

- Theoretical exercises: 8% of the total grade
- Midterm and final exams: 10% of the total grade
- Quizzes: 2% of the total grade

References

- Larry L. Peterson and Bruce S. Davie, *Computer Networks: A Systems Approach*, 5th Edition, 2011.

30 Game Theory

Course Code: 40456, Units: 3, Prerequisite: Data Structures and Algorithms, Engineering Statistics and Probability

Course Objectives

Game theory has wide applications in various fields, especially economics, business, and social sciences. Generally, in game theory, we deal with systems that involve intelligent and self-interested agents, each of whom changes the system's state according to their own interests. Game theory provides the tools to analyze such systems and helps us control them in a logical and structured way. This course aims to introduce students to the basic concepts of game theory and a few examples of its applications in modeling, mathematical analysis, and simulation.

Course Topics

1. **Normal Form Games** (4 sessions)
    - Rational Behavior and Utility Function
    - Definition of Normal Form Games
    - Simple and Mixed Nash Equilibria
    - Examples of Classic Normal Form Games
    - Methods to Compute Equilibria in Simple Normal Form Games
2. **Extensive Form Games** (2 sessions)
    - Definition of Extensive Form Games
    - Subgame Perfect Equilibrium
    - Examples of Classic Extensive Form Games
    - Methods to Compute Equilibria in Simple Extensive Form Games
3. **Evolutionary Game Theory** (2 sessions)
    - Evolutionarily Stable Strategies
    - Relationship to Nash Equilibria
    - Mixed Evolutionarily Stable Strategies
4. **Braess Paradox and Traffic Modeling using Game Theory** (1 session)

- Game Theory in Traffic Modeling
- Nash Equilibrium in Traffic Systems
- Braess Paradox

5. **Matching Markets** (2 sessions)
    - Bipartite Graphs
    - Perfect Matchings
    - Market Clearing Prices
    - Relationship to Auctions
6. **Bargaining Models** (2 sessions)
    - Modeling Human Interaction (Nash Bargaining Solution, Final Game)
    - Modeling Exchange between Two Humans (Stable Outcomes, Equilibrium Outcomes)
7. **Auction Mechanism Design** (5 sessions)
    - Definition of Auctions and Game-Theoretic Modeling
    - Types of Auctions and Their Relations (German, Japanese, English, First Price, Second Price)
    - Analysis of Second Price Auctions
    - Introduction to VCG and Sponsored Search Auctions
8. **Simple Networked Trade Models with Intermediaries** (2 sessions)
    - Pricing in Markets
    - Game-Theoretic Modeling of Trade on Intermediated Networks
    - Equilibrium Points and Relationship with Auctions
9. **Signaling Games and Information Cascades** (3 sessions)
    - Signaling Games
    - Speech-Act Theory
    - Bayes' Theorem and Decision Making under Uncertainty
    - Herding Behavior
    - Information Cascades

10. **Market Analysis, Network Effects, and Externalities** (2 sessions)
    - Market Analysis without Network Effects
    - Market Analysis with Network Effects
    - Dynamic View of Markets and Stable vs. Unstable Points
    - Positive and Negative Externalities
11. **Social Choice and Voting Mechanisms** (2 sessions)
    - Definition of Social Choice and Voting Mechanisms
    - Familiar Voting Mechanisms
    - Arrow's Impossibility Theorem
12. **Asset Valuation and Intellectual Property** (2 sessions)
    - Externalities and Coase Theorem
    - Tragedy of the Commons
    - Intellectual Property
13. **Introduction to Coalitional Game Theory** (1 session)
    - Definition of Coalitional Games
    - Core Concept
    - Solving Classic Coalitional Games
    - Shapley Value

Evaluation

- Theoretical Exercises: 20% of the total grade
- Exams (Midterm, Final, and Quizzes): 80% of the total grade

References

- Yoav Shoham and Kevin Leyton-Brown, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*, Cambridge University Press, 2008.
- David Easley and Jon Kleinberg, *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*, Cambridge University Press, 2010.
- Martin J. Osborne and Ariel Rubinstein, *A Course in Game Theory*, MIT Press, 1994.

31 Software Engineering

Course Code: 40474, Units: 3, Prerequisite: Systems Analysis and Design

Course Objectives

This course focuses on the engineering principles that must be adhered to throughout the software development process. Students have already been introduced to software construction (programming), requirements analysis, and software design in previous courses. The goal of this course is not to teach new methods for requirements analysis or software design but to emphasize the production of software as an engineering product, similar to other products produced in engineering fields. Initially, the difference between products produced through engineering methods and those produced through artistic methods will be discussed. Then, the expectations that an engineering product must meet will be clarified. The course further emphasizes engineering production methods such as modeling, measurability, evaluation, verification, and validation of intermediate products. Since students have been less familiar with formal descriptions of requirements, measurement, estimation, and testing in previous courses, these topics will receive additional focus in this course. Finally, activities such as project management, scheduling, risk management, configuration management, and quality assurance will be reviewed, emphasizing their role in producing software as an engineering product.

Course Topics

1. **Introduction** (2 sessions)
2. **Process Models** (2 sessions)
3. **Agile Development** (1 session)
4. **Understanding Requirements** (1 session)
5. **Formal Methods** (5 sessions)
6. **Design Concepts** (1 session)
7. **Architectural Design** (1 session)
8. **Interface Design** (1 session)
9. **Pattern-Based Design** (1 session)
10. **Testing Strategies** (1 session)

11. **Testing Methods** (4 sessions)
12. **Product Measurement** (1 session)
13. **Process and Project Measurement** (1 session)
14. **Estimation** (1 session)
15. **Quality Concepts** (1 session)
16. **Review Methods** (1 session)
17. **Quality Assurance** (1 session)
18. **Configuration Management** (1 session)
19. **Project Management** (1 session)
20. **Scheduling** (1 session)
21. **Risk Management** (1 session)

Evaluation

- 3 Practical-Theoretical Exercises during the term: 20% of the total grade
- 3 Multiple Choice Tests on the course materials during the term: 30% of the total grade
- Approximately 5 Short Quizzes during the term: 10% of the total grade (bonus points)
- Final Exam (Descriptive and Multiple Choice): 50% of the total grade
- Optional Seminar on topics not covered in class but related to the course content (with prior approval): 10% of the total grade (bonus points)

References

- R. S. Pressman, *Software Engineering: A Practitioner's Approach*, 8th Edition, McGraw-Hill, 2014.
- P. Ammann and J. Offutt, *Introduction to Software Testing*, Cambridge University Press, 2008.
- J. Woodcock and J. Davies, *Using Z: Specification, Refinement, and Proof*, Prentice-Hall, 1996.

32 Computer Simulation

Course Code: 40634, Units: 3, Prerequisite: Engineering Probability and Statistics

Course Objectives

The goal of this course is to familiarize students with various simulation methods and related topics.

Course Topics

1. **Introduction to Simulation**
2. **Familiarization with MATLAB or Similar Tools as Computational Tools for the Course**
3. **Basic Principles and Examples of Simulation**
4. **Discrete Event System Simulation Concepts**
5. **Several Examples of Simulation**
6. **Discrete-Event Simulation System Implementation Patterns**
7. **Types of Discrete Event Simulation System Structures**
8. **Sorted List Processing**
9. **Methods of Drawing Systems for Simulation**
10. **Statistical Models in Simulation**
11. **Brief Review of Statistics and Probability**
12. **Discrete Distributions**
13. **Continuous Distributions**
14. **Empirical Distributions**
15. **Uniform Random Number Generation**
16. **Required Characteristics for Random Numbers**
17. **Methods of Random Number Generation**

18. **Randomness Tests for Sequences**
19. **Generation of Random Variables**
20. **Inverse Transformation Method**
21. **Acceptance-Rejection Method**
22. **Combination**
23. **Convolution**
24. **Arrival Modeling**
25. **Data Collection**
26. **Evaluation of Sample Independence**
27. **Distribution Fitting from Data**
28. **Parameter Estimation**
29. **Goodness-of-Fit Testing**
30. **Model Selection in the Absence of Data Samples**
31. **Input Process Models**
32. **Verification and Validation of Simulation Models**
33. **Analysis of Output Data**
34. **Transient and Steady-State Behavior of Stochastic Processes**
35. **Types of Simulation Based on Output Analysis**
36. **Statistical Analysis of Steady-State Parameters**
37. **Design of Experiments and Sensitivity Analysis**
38. **Advanced Topics in Simulation**
39. **Monte Carlo Simulation**
40. **Real-World Examples of Simulation**

Evaluation

- Theoretical Exercises: 3 points
- Midterm and Final Exams: 15 points
- Quizzes: 2 points

References

- Banks, Carson, Nelson, and Nicol. *Discrete-Event System Simulation*, 5th Edition, Prentice-Hall, 2010.

33 Computer Engineering Project

Course Code: 40760, Units: 3, Prerequisite: Presentation of Scientific and Technical Topics

Course Objectives

The goal of the undergraduate project is to analyze, design, and implement a real project or conduct a research project based on the concepts learned throughout the undergraduate program.

34 Signals and Systems

Course Code: 40242, Units: 3, Prerequisite: Foundations of Electrical and Electronic Circuits, Co-requisite: None

Course Summary:

This course aims to familiarize students with modeling, describing, and analyzing signals and systems in both time and frequency domains from theoretical and practical perspectives. In addition to theoretical exercises, MATLAB practices are included to reinforce practical understanding of the presented concepts.

Course Outline:

- Introduction
- Continuous-time and Discrete-time Signals
- Signal Transformations
- Signal Properties and Types (periodic, even, odd, etc.)
- System Properties (memoryless, causal, stable, linear, time-invariant)
- Linear Time-Invariant (LTI) Systems
- Convolution (Continuous and Discrete)
- Impulse Response
- Linear Constant-Coefficient Differential Equations (LCCDE) and Block Diagrams
- Fourier Series for Periodic Signals
- LTI System Response to Complex Exponentials
- Fourier Series Representations for Continuous and Discrete-Time Periodic Signals
- Fourier Series Properties (linearity, time shift, time scaling, etc.)
- Fourier Series and LTI Systems: System Function and Frequency Response
- Continuous-Time Fourier Transform (CTFT)

- Fourier Transform for Aperiodic and Periodic Signals
- CTFT Properties (linearity, time shift, etc.)
- Multiplication and Convolution
- Systems Described by LCCDE
- Discrete-Time Fourier Transform (DTFT)
- DTFT for Aperiodic and Periodic Signals
- DTFT Properties (periodicity, linearity, time shift, etc.)
- Multiplication and Convolution
- Systems Described by LCCDE
- Time/Frequency Description of Signals and Systems
- Magnitude and Phase of the Fourier Transform
- Frequency Response Magnitude and Phase
- Log Magnitude Plots
- Bode Plots
- Ideal and Non-Ideal Filters
- First and Second Order Continuous and Discrete-Time Systems
- Sampling
  - Sampling Theorem, Impulse Train, Interpolation, Aliasing
- Laplace Transform
  - Region of Convergence, Inverse Transform, Pole-Zero Diagram
  - Properties (linearity, time shift, etc.)
  - Time and s-domain Differentiation and Integration
  - Initial and Final Value Theorems
  - Causality and Stability
  - Systems Described by LCCDE
  - Butterworth Filter, Block Diagram Representation

  - One-Sided Laplace Transform

- z-Transform
  - Region of Convergence, Inverse Transform, Pole-Zero Diagram
  - Properties (linearity, time shift, etc.)
  - Initial Value Theorem
  - Causality and Stability
  - Systems Described by LCCDE
  - Block Diagram Representation
  - One-Sided z-Transform

Books Used:

- Alan V. Oppenheim, Alan S. Willsky, and S. Hamid Nawab, *Signals and Systems*, 2nd Edition, Prentice Hall, 1996.

Evaluation:

- Exercises: 15%
- Mid-term Exam: 35%
- Final Exam: 50%

35 Modern Information Retrieval

Course Code: 40324, Units: 3, Prerequisite: Data Structures and Algorithms, Co-requisite: None

Course Summary:

This course introduces information retrieval systems. It starts with indexing operations and the Boolean retrieval model. Then the vector space model and tf-idf representation are discussed, followed by techniques for speeding up document scoring and ranking. Probabilistic retrieval models are introduced along with concepts such as document classification, clustering, and learning to rank. The course continues with an introduction to web search engines and their key components such as crawlers, link graph analysis, and near-duplicate detection. Finally, recommender systems and advanced topics in information retrieval are covered.

Course Outline:

- Introduction to Information Retrieval
- Boolean Information Retrieval Systems and Indexing
- Document Preprocessing: Text Operations and Word Normalization
- Tolerant Retrieval
  - Wild-card Queries, Spelling Correction
- Blocked and Distributed Indexing
- Map-Reduce
- Index Compression
  - Dictionary Compression
  - Posting List Compression: Byte-wise and Gamma Encoding
- Vector Space Model and tf-idf Representation
- Document Scoring and Ranking (Efficiency Improvements)
- Evaluation of Information Retrieval Systems and Metrics
- Probabilistic Information Retrieval Models

- Language Models
- Document Classification
  - Naïve Bayes Classifier, Linear Classifiers
- Document Clustering
  - k-means, Hierarchical Clustering
- Learning to Rank
- Dimensionality Reduction and Word Embeddings
  - Latent Semantic Indexing (LSI), Word2Vec
- Web Search Engines
  - Crawlers, Near-Duplicate Detection
  - Link Graph Analysis and PageRank
- Recommender Systems
  - Content-Based Methods
  - Collaborative Filtering
  - Hybrid Approaches
- Advanced Topics
  - Personalization, Information Retrieval in Social Networks
  - Question Answering Systems
  - Sentiment Analysis
  - Cross-Lingual Information Retrieval

Books Used:

- C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*, Cambridge University Press, 2008.

Evaluation:

- Mid-term Exam: 25%
- Final Exam: 35%

- Project: 25%
- Short Exams: 10%
- Quizzes: 5%

36 Multimedia Systems

Course Code: 40342, Units: 3, Prerequisite: Signals and Systems, Co-requisite: None

Course Summary:

This course introduces students to the fundamental concepts of multimedia and multimedia systems, considering emerging value-added services. Topics include signal processing, compression, multimedia data types, system architectures, and network protocols for multimedia applications.

Course Outline:

- Introduction to Multimedia (2 sessions)
  - Multimedia and Multimedia Systems
  - Hypermedia
  - Characteristics, Challenges, and Components of Multimedia Systems
  - Multimedia Data, Projects, and Research Topics
- Review of Signals and Systems (4 sessions)
  - Discrete-Time Signals and Systems
  - Sampling Theory
  - Scalar and Vector Quantization
  - Transform Domain Analysis
  - FFT, STFT, and Wavelet Transforms
- Audio (3 sessions)
  - Audio Representation and Playback
  - Sampling and Quantization
  - Standards and Formats
  - Temporal and Frequency Masking
  - Audio Signal Processing
  - Audio Compression
- Entropy Coding (3 sessions)

  - Lossy and Lossless Compression
  - Run-Length Encoding
  - Fixed-Length and Variable-Length Coding
  - Huffman Coding
  - Lempel-Ziv-Welch Coding
  - Arithmetic Coding

- Image (4 sessions)
  - Color Spaces: YUV, RGB, HSV, CMYK
  - Image Acquisition and Display
  - Image Enhancement
  - Image Compression: DCT, MPEG

- Video (4 sessions)
  - Basics of Analog and Digital Video
  - Video Compression
  - Intra-frame and Inter-frame Coding
  - Motion Estimation and Compensation
  - Video Quality Evaluation
  - Video Coding Standards: MPEG1, MPEG2, MPEG4, H.261, H.263, H.264

- Multimedia Systems (4 sessions)
  - Standalone vs. Networked Systems
  - Orchestrated vs. Live Systems
  - System Components
  - Real-Time Multimedia System Architecture

- Multimedia Networking (3 sessions)
  - Multimedia Data Delivery Quality
  - Streaming Protocols
  - Error Concealment
  - Priority Encoding

  - Overlay Networks
  - Packet Loss, Congestion, and QoS
  - Unicasting and Multicasting
  - Wireless Multimedia
- Multimedia Applications (3 sessions)
  - Internet Telephony
  - Digital Video Broadcasting
  - IPTV and Interactive Television
  - E-learning
  - Human-Computer Interaction
  - Multimedia Home Platforms
  - Multimedia Information Retrieval Systems
  - 3D Technologies

Books Used:

- R. Steinmetz and K. Nahrstedt, *Multimedia: Computing, Communications and Applications*, Prentice Hall, 1995.
- R. Steinmetz and K. Nahrstedt, *Multimedia Fundamentals: Media Coding and Content Processing*, Prentice Hall, 2002.
- K. R. Rao, Z. S. Bojkovic, and D. A. Milanovic, *Multimedia Communication Systems: Techniques, Standards and Networks*, Prentice Hall, 2002.

Evaluation:

- Theoretical Assignments: 3 points
- Mid-term and Final Exams: 12 points
- Quizzes: 3 points

37 Data Transmission

Course Code: 40343, Units: 3, Prerequisite: Signals and Systems, Co-requisite: None

Course Summary:

The goal of this course is to familiarize students with how data is transmitted through various media and by different methods, as well as the challenges and issues associated with each.

Course Outline:

- Transmission Media (6 sessions)
  - Twisted Pair, Shielded Twisted Pair
  - Coaxial Cable
  - Waveguide
  - Optical Fiber
  - Free-Space Optical Link
  - Microwave Link
  - Satellite
- Sources of Errors (4 sessions)
  - Thermal Noise
  - Electrical Noise (EMI, RFI)
  - Attenuation Distortion
  - Delay Distortion
  - Echo
  - Harmonic Distortion
  - Intermodulation Distortion
  - Crosstalk
  - Fading
- Error Detection and Correction (3 sessions)
  - Longitudinal Redundancy Check (LRC)
  - Vertical Redundancy Check (VRC)

  - Two-Dimensional Parity Check (VRC-LRC)
  - Cyclic Redundancy Check (CRC)
  - Checksum
  - Hamming Code
- Types of Modulation (4 sessions)
  - Analog Modulation
  - Digital Modulation
  - Pulse Modulation
- Multiplexing (2 sessions)
  - Time Division Multiplexing (TDM)
  - Frequency Division Multiplexing (FDM)
  - Code Division Multiplexing (CDM)
- Multiple Access (2 sessions)
  - Time Division Multiple Access (TDMA)
  - Frequency Division Multiple Access (FDMA)
  - Code Division Multiple Access (CDMA)
- Channel Capacity (2 sessions)
  - Shannon's Theorem
  - Optimal Power Allocation
- Data Compression (3 sessions)
  - Audio Compression
  - Huffman Coding
  - Compression in Fax Systems
- Switching (1 session)
  - Circuit Switching
  - Message Switching
  - Packet Switching

- Flow Control Efficiency (3 sessions)
  - Stop-and-Wait Method
  - Sliding Window Protocol
  - Error Impact on Efficiency

Books Used:

- W. Stallings, *Data and Computer Communications*, Prentice-Hall, 1996.
- F. Halsall, *Data Communications, Computer Networks, and Open Systems*, 4th Edition, Addison Wesley, 1996.
- A. S. Tanenbaum, *Computer Networks*, 3rd Edition, Prentice-Hall, 1996.
- Edham Sadeghi (Translator), *Principles of Data Communication*, Tizhooshan-e-Sarzamin-e-Kohan Publishing, 2005. (in Persian)

Evaluation:

- Theoretical Assignments: 4 points
- Mid-term and Final Exams: 16 points

38 Fundamentals of 3D Computer Vision

Course Code: 40344, Units: 3, Prerequisite: Linear Algebra or Engineering Mathematics, Co-requisite: None

Course Summary:

This course introduces students to fundamental concepts and methods for analyzing images to achieve a high-level understanding of their content. Topics include image formation and color representation, basic signal and image processing, 3D geometry, feature extraction, robust model fitting, clustering and segmentation, object recognition, nearest neighbor search, and deep learning techniques in computer vision.

Course Outline:

- Signal and Image Processing
  - Basic Signal Processing Concepts
  - Overview of Signals and Systems
  - Convolution Function
  - Fourier Transform
  - Image Filtering
- Basics of 3D Geometry
  - Basic Geometric Concepts
  - Brief Review of Linear Algebra
  - Parametrization of Rotation Matrices
  - Homogeneous Coordinates
  - Pinhole Camera Model
  - Mapping from Meters to Pixel Coordinates
- Cameras and Projections
  - Parallel and Perspective Projection
  - Single-Axis Camera Rotations
  - Simple Mosaic Image Construction
  - Intrinsic and Extrinsic Camera Parameters

  - General Camera Motion and Linear Mapping Estimation
  - Camera Translation
- 3D Reconstruction from Stereo Vision
  - Surface Reconstruction and Rendering
  - Point Cloud Triangulation
  - Surface Mapping, Image-Based Rendering
  - Planar Surfaces and Linear Homographies
- Multi-Camera Systems
  - Perspective Cameras and Bundle Adjustment
  - Parallel Projection Cameras
  - Camera Calibration and 3D Coordinate Systems
  - Affine Structure
- Keypoint Extraction
- Robust Model Fitting
- Clustering and Segmentation
  - Graph Cuts
- Object Recognition
  - Template Classification
  - Nearest Neighbors, PCA, Dimensionality Reduction
  - Naive Bayes Classifier
  - Ensemble of Simple Classifiers
  - Neural Networks
- Nearest Neighbors
- Deep Learning in Computer Vision
  - Neural Networks and Backpropagation
  - CNN Architectures: New Ideas, Advantages, and Limitations
  - Spatio-temporal Deep Networks

– Training Deep Neural Networks using PyTorch

Books Used:

- Stefan Carlsson, *Geometric Computing in Image Analysis and Visualization*, Lecture Notes, KTH University, 2007.
- Richard Szeliski, *Computer Vision: Algorithms and Applications*, 1st Edition, Springer, 2010.

Evaluation:

- Final Exam: 40%
- Mid-term Exam: 15%
- Quizzes: 10%
- Assignments: 15%
- Final Project: 20%

39 Information Technology Ethics

Course Code: 40347, Units: 3, Prerequisite: None, Co-requisite: None

Course Summary:

This course explores the ethical, legal, and social implications of information technology. It emphasizes the importance of professional conduct and ethical behavior in the development, deployment, and use of IT systems. Topics include ethical and legal frameworks, professional codes of conduct, privacy, security, intellectual property, digital rights, cybercrimes, social responsibility, and the philosophical underpinnings of ethics in the digital world.

Course Outline:

- Introduction (2 sessions)
  - Course goals, structure, and teaching model
  - Principles of ethics in engineering and IT
- Relationship between ethics, etiquette, and law (1 session)
  - Comparison between ethics and etiquette
  - Compatibility and distinction with legal norms
  - Law as retrospective, etiquette as forward-looking
- History and types of ethical systems (5 sessions)
  - Western ethical philosophies
  - Islamic and Iranian ethics
  - Global ethics, golden rules, and customary norms
- Ethical dilemmas and resolution frameworks (3 sessions)
  - Ethical conflicts and ambiguities
  - Ethical judgment models and decision processes
- Nature and application of professional codes (2 sessions)
  - Charters, bylaws, and conduct codes
  - Structure and evaluability of ethical documents

- Models for code generation and refinement (2 sessions)
  - Organizational and group-level codes
  - Success metrics and code improvement models
- Professional ethics and engineering systems (2 sessions)
  - Professional engineering ethics in IT
  - National and global engineering systems
- Intellectual property in IT (3 sessions)
  - Financial and creative rights
  - Copyright, patents, and legal protections
  - Future of digital rights and ethical frameworks
- Ethics in the information and virtual society (3 sessions)
  - Free information flow, citizen rights, and transparency
  - Privacy, virtual reality, and social media ethics
  - Democracy, digital health literacy, and public awareness
- Cybercrimes and IT-related offenses (4 sessions)
  - Malware, intrusion methods, and cyber threats
  - Internet police, cybercrime laws, and social security
  - Organizational security, open-source licensing, and privacy concerns
  - Green IT and sustainable practices
- Emerging issues in IT ethics (2 sessions)
  - Network neutrality, augmented/virtual reality, and infosphere philosophy
  - NBIC ethics and the Fourth Revolution (Floridi)

Books Used:

- George Reynolds, *Ethics in Information Technology*, Thomson, 2011.
- Luciano Floridi, *The Fourth Revolution: How the Infosphere is Reshaping Human Reality*, Oxford University Press, 2014.

- Luciano Floridi, *Information and Computer Ethics*, Cambridge University Press, 2010.
- Ibo van de Poel, *Ethics, Technology, and Engineering*, Wiley-Blackwell, 2011.
- Harris, M. J. Rabins, and C. E. Harris, *Engineering Ethics: Concepts & Cases*, Thomson, 2004.
- M. W. Martin, *Ethics in Engineering*, McGraw-Hill, 2005.
- Duncan Langford, *Internet Ethics*, Macmillan Press Ltd, 2000.

Evaluation:

- Exercises (Comprehension and Problem Solving): 6 points
- Midterm and Final Exams: 12 points
- Quizzes: 2 points

40 VLSI Design

Course Code: 40353, Units: 3, Prerequisite: Digital System Design, Basic Electrical and Electronic Circuits, Co-requisite: None

Course Summary:

This course introduces the design, analysis, and fabrication of VLSI (Very Large-Scale Integration) systems with a focus on transistor-level design. It covers the principles of modern VLSI circuit design, including MOS transistor modeling, logic gate implementation, power and delay analysis, and layout planning.

Course Outline:

- Introduction to VLSI Circuits
- VLSI Circuit Benchmarks and Abstraction Levels
- Chip Fabrication Process and Photolithography
- Layout Process and Design Rules
- Fabrication Defects and Manufacturing Issues
- Stick Diagrams
- MOS Transistor Characteristics
  - I-V Characteristics of nMOS and pMOS
  - DC Response, Body Effect, and Channel Length Modulation
  - Leakage and Subthreshold Currents
  - Latch-up Effect and Parasitics
- On-chip Interconnects
  - Wire Capacitance and Resistance
  - Routing and Vias
- Logic Gate and Combinational Circuit Design
  - CMOS Logic, Pseudo-nMOS, Domino Logic
  - Switch Logic, DCVS Logic

- Power Consumption Analysis
  - Static and Dynamic Power
- Delay Calculation
  - Logical Effort, Branch Effort, Path Delay
- Sequential Elements
  - Static and Dynamic Latches
  - Clocking Structures
- Arithmetic and Logic Elements
  - Adders, Multipliers, Shifters, ALU
- Floorplanning
- VLSI Circuit Testability

Books Used:

- Wayne Wolf, *Modern VLSI Design: System-on-Chip Design*, 3rd Edition, 2004.
- J. M. Rabaey, A. Chandrakasan, and B. Nikolić, *Digital Integrated Circuits: A Design Perspective*, 2005.
- N. H. E. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 3rd Edition, Addison-Wesley, 2005.

Evaluation:

- Theoretical Exercises: 3 points
- Midterm and Final Exams: 15 points
- Quizzes: 2 points

41 Design of Algorithms

Course Code: 40354, Units: 3, Prerequisite: Data Structures and Algorithms, Co-requisite: None

Course Summary:

This course introduces common strategies for designing efficient algorithms for a variety of computational problems. Emphasis is placed on analyzing algorithm efficiency and proving correctness. Topics include classic algorithmic paradigms, complexity theory, graph algorithms, network flows, and approximation techniques.

Course Outline:

- Introduction and Sample Problems
  - Solvability, Algorithm Analysis, Running Times
  - Longest Contiguous Subsequence, 3-SUM Problem
- Inductive Algorithms
  - Polynomial Evaluation, One-to-One Mapping, Celebrity Problem
- Divide and Conquer
  - Exponentiation, Recurrences, Closest Pair
  - Strassen's Matrix Multiplication, Fast Fourier Transform
- Greedy Algorithms
  - Coin Change, Scheduling Problems, Fractional Knapsack
  - Huffman Coding, Stable Matching (Gale-Shapley)
- Dynamic Programming
  - Fibonacci, Weighted Interval Scheduling, Coin Change
  - Matrix Chain Multiplication, Knapsack, Sequence Alignment
  - Longest Common Subsequence, Longest Increasing Subsequence
  - Maximum Weight Independent Set on Trees, Optimal BST
- State Space Search
  - Backtracking: Eight Queens, Subset Sum

  - Branch and Bound: TSP, Game Trees, Alpha-Beta Pruning

- Graph Algorithms
  - MST: Kruskal and Prim
  - Fibonacci Heaps, Amortized Analysis for Decrease-Key
  - All-Pairs Shortest Paths: Floyd-Warshall, Johnson

- String Matching
  - Fingerprinting and Rabin-Karp
  - Finite Automata and Knuth-Morris-Pratt (KMP)

- Network Flows
  - Max-Flow and Min-Cut: Ford-Fulkerson
  - Improvements: Edmonds-Karp
  - Applications: Bipartite Matching, Disjoint Paths, Matrix Rounding

- Linear Programming
  - Standard Formulation, Modeling with LP
  - Simplex Algorithm

- Computational Complexity
  - Polynomial-Time Reductions, Satisfiability
  - NP Class, NP-Completeness, Cook's Theorem
  - Hamiltonian Cycle, Graph Coloring, Subset Sum

- Approximation Algorithms
  - Vertex Cover, TSP, Approximation Hardness
  - PTAS, Knapsack Problem

Books Used:

- J. Kleinberg and E. Tardos, *Algorithm Design*, Addison Wesley, 2005.
- T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd Edition, MIT Press, 2009.

- U. Manber, *Introduction to Algorithms: A Creative Approach*, Addison-Wesley, 1989.
- G. Brassard, P. Bratley, *Algorithmics: Theory and Practice*, Prentice-Hall, 1988.

Evaluation:

- Three Theoretical Assignments: 3 points
- Three Programming Assignments: 3 points
- Midterm Exam: 7 points
- Final Exam: 7 points
- ACM-style Programming Contest: 1+ bonus point

42 Industrial Automation Lab

Course Code: 40401, Units: 1, Prerequisite: Computerized Measurement and Control, Co-requisite: None

Course Summary:

This laboratory course aims to provide students with practical familiarity with tools, equipment, and software used in industrial automation. Students will be able to understand and, if necessary, design, implement, or improve automated processes in production, assembly, packaging, monitoring, and quality control environments.

Course Outline:

- Actuators such as various motors, servomotors, drivers, valves, relays, and switches
- Installation and setup of LabView software and introduction to its environment
- Building and testing a TCP connection in LabView
- Designing and implementing a three-floor elevator simulation in LabView
- Introduction to LogoSoft software
- Familiarization with one or more commonly used industrial Programmable Logic Controllers (PLC) and ladder logic programming; use of analog and digital I/O interface boards based on industrial PCs
- Implementation of a four-phase traffic light system using a traffic light training board and PLC
- Designing and implementing a tank mixer system on the training board and PLC
- Designing and implementing an elevator system in LogoSoft
- Higher-level automation programming using Step 7 or Grafcet programming paradigms

Books Used:

- G. Dunning, *Introduction to Programmable Logic Controller*, 3rd Edition, Thompson, 2017.

- F. D. Petruzella, *Programmable Logic Controllers*, McGraw-Hill Education, 5th Edition, 2016.
- C. T. Jones, *STEP 7 programming made easy in LAD, FBD, and STL: A practical guide to programming S7-300/S7-400 Programmable Logic Controllers*, Patrick-Turner Publishing, 2013.
- R. D. Rosandich, *Fundamentals of Ladder Diagram Programming*, EC & M Books, 1999.
- J. Ravis and J. Kring, *LabVIEW for Everyone: Graphical Programming Made Easy and Fun*, 3rd Edition, Prentice Hall, 2006.

Evaluation: (Not specified)

43 VLSI Lab

Course Code: 40402, Units: 1, Prerequisite: None, Co-requisite: VLSI Design

Course Summary:

This laboratory course aims to familiarize students with automated tools for designing and analyzing digital chips. Students will apply the concepts learned in the VLSI Design course using these tools for practical experiments.

Course Outline:

- Implementation of an inverter and analysis of its circuit characteristics
- Design and simulation of a 4-bit counter using HSpice
- Implementation of a NOR3 gate in three logics: Static CMOS, Pseudo-NMOS, and Domino Logic, and their comparison
- Gate sizing to optimize the speed of a path
- Designing a gate using layout drawing tools and verifying its correctness
- Introduction to the synthesis tool Design Compiler and synthesis of a 16-bit multiplier for speed and area optimization
- Power consumption calculation using Power Compiler tool and applying Clock Gating; comparison with previous power consumption
- Introduction to automatic layout design tool SOC Encounter and layout design of a simple sequential circuit
- Layout design of the circuit
- Automatic layout design of a 16-bit multiplier and functional verification using Modelsim and Hsim

Books Used:

- Wayne Wolf, *Modern VLSI Design: IP-Based Design*, 4th Edition, Prentice-Hall, 2009.

Evaluation: (Not specified)

44 Biology Laboratory

Course Code: 40409, Units: 1, Prerequisite: None, Co-requisite: None

Course Summary:

This course covers the principles and methods of molecular and cellular biology techniques. Students will gain practical experience studying cell structure, organelles, and various methods to examine cellular behavior and processes.

Course Outline:

- Introduction to skills in molecular and cellular biology laboratory
- Study of the function and components of the optical microscope, structural overview, general microscopy techniques, applications, and functions of research and advanced microscopes
- Examination of cell structure and function in samples of unicellular organisms and animal and plant cells (using optical microscope)
- Measurement of cell dimensions and microscopic samples
- Cell counting in suspension media
- Cellular staining and organelles such as mitochondria and lysosomes in cells
- Study of mitosis cell division process and observation of its stages
- Familiarization with permanent slide preparation from plant and animal tissues
- General staining of nucleus and cytoplasm
- Plasmid DNA isolation, PCR, sequence result analysis, cell culture database searching, DNA transfection into cells, and gene expression study

Books Used: (Not specified)

45 Advanced Logic Design

Course Code: 40412, Units: 3, Prerequisite: Logic Circuits, Co-requisite: None

Course Summary:

This course introduces students to the concepts of synchronous and asynchronous circuit design and timing hazards, with a focus on advanced digital circuit design considerations such as testability and power consumption.

Course Outline:

- Review of sequential circuits
- Design and simplification of synchronous sequential circuits
- Asynchronous sequential circuits
- Timing delays and types of hazards
- Multivalued and mixed logic
- Design considerations for testability and low power in modern designs

Evaluation:

- Theoretical Exercises: 3 points
- Mid-term and Final Exams: 15 points
- Quizzes: 2 points

Books Used:

- B. J. LaMeres, *Introduction to Logic Circuits & Logic Design with VHDL*, 2nd Edition, Springer, 2019.
- T. Ndjountche, *Digital Electronics Vol. 2, Sequential and Arithmetic Logic Circuits*, Wiley, 2016.
- Ch. H. Roth and L. L. Kinney, *Fundamentals of Logic Design*, 7th Edition, Cengage Learning, 2013.
- V. G. Oklobdzija, V. M. Stojanovic, D. M. Markovic, and N. M. Nedovic, *Digital System Clocking: High-Performance and Low-Power Aspects*, Wiley, 2003.

- J. F. Wakerly, *Digital Design Principles & Practices*, Prentice Hall, 2001.
- M. M. Mano, Ch. R. Kime, and T. Martin, *Logic & Computer Design Fundamentals*, 5th Edition, Prentice Hall, 2006.
- Alireza Ejlali, *Logic Circuits*, 1st Edition, Nasir Publishing, 2018.

46 Web Programming

Course Code: 40419, Units: 3, Prerequisite: Advanced Programming, Co-requisite: None

Course Summary:

This course aims to familiarize students with the fundamental concepts and principles of web software design. Students will learn about client-side and server-side programming and their interaction, as well as become acquainted with a widely used framework for web application development.

Course Outline:

- Introduction (1 session)
  - Course overview, web history, HTTP protocol
- Page Design (2 sessions)
  - HTML structure, elements and attributes, paragraphs, formatting, links, lists
  - Images, tables, forms, new HTML5 elements
- Styling (2 sessions)
  - CSS definition, formatting, selectors, inheritance and cascade, design principles
  - Page layout, box model, floats, positioning, pseudo-classes
- JavaScript (4 sessions)
  - Language structure, uses, statements and functions, variables and data types, control structures
  - Arrays, objects, object definitions, constructors, data encapsulation
  - DOM model, editing elements and styles, event handling, exceptions
  - jQuery library, selectors, events, effects and animations
- Data Storage (2 sessions)
  - Introduction to XML, uses, DTD, XSLT transformation, introduction to JSON

  - Relational databases, database creation, SQL query language

- Server Interaction (2 sessions)
  - CGI interface, GET and POST submission, form processing, cookies
  - AJAX usage, request sending, response parsing, applications

- Python (5 sessions)
  - Language structure, operators, data types, lists, strings, tuples, dictionaries
  - Functions, modules, packages, anonymous functions, variable arguments, decorators
  - Classes and objects, constructors, inheritance, exception handling
  - Files, text processing, regular expressions, applications
  - Web page reading, Python web server, introduction to WSGI

- Web Architecture (2 sessions)
  - Layering, client-server architecture, three-tier architecture, MVC architecture
  - Data models, types of relations, mapping to relational databases

- Django Framework (6 sessions)
  - Basic concepts, installation and setup, components, overall architecture
  - Project creation, database definition, admin setup, adding views
  - Model layer, object-relational mapping, inheritance, query execution
  - View layer, URL mapping, request and response objects, generic views
  - Template layer, template language, built-in tags and filters
  - Form processing, built-in widgets, validation

- Advanced Topics (4 sessions, if time permits)
  - Middleware, optimization, compression, caching usage
  - Authentication, access control, user and group management
  - Security, protection against attacks, encryption
  - Sessions, session state storage, hybrid methods
  - Internationalization, localization, translation tools

Evaluation:

- Practical Exercises: 5 points
- Project: 5 points
- Midterm Exam: 4 points
- Final Exam: 6 points

Books Used:

- S. M. Schafer, *HTML, XHTML, and CSS Bible*, 5th Edition, Wiley Publishing, 2010.
- J. Forcier, P. Bissex, and W. Chun, *Python Web Development with Django*, Pearson Addison-Wesley, 2009.
- W. J. Chun, *Core Python Applications Programming*, 3rd Edition, Pearson Addison-Wesley, 2012.
- M. Fowler, D. Rice, M. Foemmel, E. Hieatt, R. Mee, and R. Stafford, *Patterns of Enterprise Application Architecture*, Pearson Addison-Wesley, 2003.

47 Information Technology Project Management

Course Code: 40428, Units: 3, Prerequisite: None, Co-requisite: None

Course Summary:

IT professionals in managerial and executive roles deal with projects that combine software, hardware, communication, and information components, often involving multiple teams. Managing these projects is challenging and even more complex with outsourcing. This course introduces students to advanced project management concepts specific to IT projects, including software project management within the broader scope of IT project management. Students will learn to function as executors, clients, consultants, or supervisors throughout the project lifecycle. Additionally, the course builds skills in using common project management tools and software to effectively manage IT projects.

Course Outline:

- Introduction (2 sessions)
  - Objectives, syllabus overview, teaching model and framework
  - Fundamental management concepts
- Overview of IT Project Management (1 session)
- Business Cases (2 sessions)
- Project Charter (2 sessions)
- Project Team (2 sessions)
- Scope Management Plan (2 sessions)
- Work Breakdown Structure (2 sessions)
- Scheduling and Budgeting Projects (2 sessions)
- Project Management Software, Websites, and Dashboards (1 session)
- Project Management Body of Knowledge (PMBOK) Standards (1 session)
- Risk Management Plan (2 sessions)

- Communication Management Plan (2 sessions)
- Quality Management for IT Projects (1 session)
- Change Management, Resistance, and Conflict Resolution (2 sessions)
- Procurement and Outsourcing Management (1 session)
- Leadership and Project Etiquette (2 sessions)
- Project Implementation and Closure Plan (1 session)
- Maturity Models and Agile Methods in IT Project Management (1 session)

Evaluation:

- Skill Exercises (Simulated Project Management Activities): 6 points
- Midterm and Final Exams: 12 points
- Quizzes: 2 points

References:

- Jack T. Marchewka, *Information Technology Project Management*, WILEY, 2014.

48 Multicore Computing

Course Code: 40432, Units: 3, Prerequisites: Advanced Programming, Computer Architecture, Co-requisite: None

Course Objectives:
The main objective of this course is to familiarize students with the architecture of multicore and manycore systems and parallel programming for these systems. The course begins with an overview of architecture, fundamental concepts, and challenges of multicore and manycore systems, followed by introduction to tools and methods for parallel programming on various multicore and manycore platforms.

Course Outline:

- Introduction to multicore system architecture and parallel programming models
- History of multicore systems
- Challenges of efficient programming on multicore systems
- Levels of parallelism in programs
- Performance acceleration analysis on homogeneous and heterogeneous multicore systems
- Examples of real multicore systems
- Shared-memory multiprocessors
  - Architecture overview
  - Cache coherence problem and solutions
  - Programming models and thread synchronization
  - Handling critical sections
  - Ideas to improve parallel programs
  - Common parallel computation and data management patterns
  - Computational patterns: Map, Reduction, Scan, Stencil, Recurrence, Fork-Join
  - Data management patterns: Gather, Scatter, Pack, Geometric Decomposition and Partitioning
- General parallel programming on multicore systems

  - Programming with Pthreads
  - Programming with OpenMP
- Parallel programming on vector systems
  - Overview of vector and array systems
  - Intel SIMD ISA introduction
  - CELL BE architecture and programming
- Parallel programming on general-purpose graphics processors (GPGPU)
  - Overview of GPU architecture
  - NVIDIA GPU architectures overview
  - CUDA programming
  - NVIDIA Profiler introduction
- Introduction to parallel programming in distributed systems
  - MPI library introduction and message-passing programming model

Evaluation:

- Theoretical Exercises: 3 points
- Midterm and Final Exams: 15 points
- Quizzes: 2 points

References:

- D. A. Patterson and J. L. Hennessy, *Computer Architecture: A Quantitative Approach*, Morgan Kaufmann, 2019.
- J. Sanders and E. Kandrot, *CUDA by Examples: An Introduction to GPGPU Programming*, Addison-Wesley, 2011.
- D. B. Kirk and W. W. Hwu, *Programming Massively Parallel Processors: A Hands-on Approach*, NVIDIA, 2010.
- M. McCool, A. D. Robison, and J. Reinders, *Structured Parallel Programming*, Elsevier, 2012.

49 Computer Graphics

Course Code: 40447, Units: 3, Prerequisite: None, Co-requisite: Algorithm Design

Course Objectives:
This course introduces students to fundamental concepts of computer graphics, with a strong emphasis on 3D computer graphics, transformations and projections in 3D, lighting, coloring graphical scenes, and computer games using OpenGL. OpenGL is used within high-level programming languages such as C, C++, and Java. Students are expected to be proficient in at least one of these languages and will learn to use OpenGL throughout the semester.

Course Outline:

- Introduction to basic concepts and graphics hardware
- 3D geometric transformations
- 3D affine transformations
- 3D object rendering
- Viewing concepts
- Steps to produce a graphical scene
- Coordinate systems
- Projection transformations: perspective, parallel, and oblique
- Rendering 3D curved and triangulated surfaces
- Introduction to spline functions and their applications
- Cubic and quartic spline functions including Bézier, B, Beta, and rational splines
- Rendering a spline using other spline functions
- Blob objects, axial rendering, and methods based on well-defined geometric shapes
- Octrees
- Binary space partition trees (BSP)

- Visible surface determination methods
- Classification, introduction, and comparison of algorithms
- Phong lighting model, lighting methods, and surface rendering techniques
- Fast rendering algorithms
- Texture mapping and surface detailing
- Haar models and their applications
- Global illumination and shaders
- Introduction to fractal geometry for objects and scenes not describable by Euclidean geometry
- Data visualization
- Computer animation
- Traditional animation methods
- Animation sequence design
- General animation functions
- Keyframe systems
- Calculating displacements and motion at varying speeds
- Camera path calculation
- Motion capture techniques for full-body and facial motion and their applications in animation, films, and games
- Introduction to computer game development
- Main elements including static (background) and moving objects, physics
- Texture application on objects
- Artificial intelligence, scenarios, game types, and music
- Introduction to game engines and their features
- Game production management

- Testing game development stages and market release

Evaluation:

- First midterm exam: 2.5 points
- Second midterm exam: 2.5 points
- Final exam: 5 points
- Programming assignments: 10 points

References:

- Hearn and Baker, *Computer Graphics with OpenGL*, 4th Edition, Prentice Hall, 2011.
- Steve Marschner and Peter Shirley, *Fundamentals of Computer Graphics*, 4th Edition, CRC Press, 2016.
- Edward Angel, *OpenGL, A Primer*, Addison Wesley, 2002.

50 Interface Circuits

Course Code: 40433, Units: 3, Prerequisite: Computer Architecture, Co-requisite: None

Course Objectives:
This course introduces students to various physical interfaces between computer systems and other systems or real environments (analog or peripheral). Students learn the communication protocols, advantages, disadvantages, applications, and design principles of these interfaces to be able to:

1. Gain relative mastery of the design principles of each introduced interface and understand their operation comprehensively.
2. Choose the correct method of connection between two or more computer systems or between a computer system and its analog peripheral environment, depending on the application environment, to plan for information transfer within or between systems.
3. Achieve a more complete understanding of a system's architecture, components, and interconnections (e.g., in studying an industrial control system for redesign purposes).

Course Outline:

- Fundamental concepts of information exchange
- Signal characteristics and transmission lines in computer systems
- Bandwidth, data rate, compression, coding, and technology constraints
- Principles of serial and parallel interface communication
- Synchronous and asynchronous interface circuits
- Servicing and addressing methods in interface circuits
- Internal system buses
- Processor buses and memory devices (Hard and On-board Memory)
- Peripheral device buses
- Interface circuits in embedded and industrial systems
- Inter-system computer buses

- USB interface (data transfer)
- HDMI interface (user interface)
- Overview of wireless interfaces:
  - Bluetooth
  - Wireless USB
  - Zigbee
- Software-hardware interfaces (Device drivers) and embedded/real-time operating system usage
- Overview of analog interfacing and ADC/DAC converters, sensors, actuators, electromagnetic interference, crosstalk, grounding, and analog-digital interface design considerations
- Practical example of interface circuits based on Raspberry Pi or similar microcontroller boards

Evaluation:

- Theoretical exercises: 3 points
- Midterm and final exams: 15 points
- Quizzes: 2 points

References:

- Jonathan W. Valvano, *Embedded Microcomputer Systems: Real Time Interfacing*, 3rd Edition, Cengage Learning, 2011.
- Gourab Sen Gupta and Subhas Chandra Mukhopadhyay, *Embedded Microcontroller Interfacing, Designing Integrated Projects*, Springer, 2010.
- Stuart R. Ball, *Analog Interfacing to Embedded Microprocessor Systems*, Elsevier, 2004.

51 Theory of Computation

Course Code: 40455, Units: 3, Prerequisite: Data Structures and Algorithms, Co-requisite: None

Course Objectives:
This course aims to introduce students to the theoretical foundations of computation, main concepts of computability models, solvable problems, mathematical logic, and an introduction to automata theory over infinite string or tree inputs. It provides the theoretical basis for students pursuing graduate studies in computation theory, algorithms, formal methods in software engineering, system verification, and lays the mathematical logic foundation needed for artificial intelligence.

Course Outline:

1. **Computability Theory and Introduction to Computational Complexity**
    - Turing machine model, Church-Turing thesis, decidable (recursive) and recognizable (recursively enumerable) languages and functions, uncomputable functions, halting problem, universal Turing machine, multitape and nondeterministic Turing machines and their equivalence (3 lectures)
    - Proof techniques for undecidability and non-recognizability including reduction from the halting problem and many-one reductions (2 lectures)
    - Introduction to other computational models (2 lectures)
        - Random Access Machine (RAM) model (von Neumann architecture)
        - Kleene recursive functions
        - Lambda calculus (Church)
        - Post systems
        - Recursion theorem and self-reference (1 lecture)
    - Computational complexity of information and string complexity (2 lectures)
    - Introduction to complexity theory, overview of time and space complexity classes and hard problems (3 lectures)
2. **Mathematical Logic from Computation Theory Perspective**

- Propositional logic: syntax, semantics, axiomatic systems, soundness and completeness theorems, decidability of propositional logic (2 lectures)
- First-order logic: syntax, semantics, compactness theorem, Löwenheim–Skolem theorem (2 lectures)
- Axiomatic system of first-order logic and its soundness theorem (1 lecture)
- Gödel's completeness theorem for first-order logic (1 lecture)
- Church's theorem on the undecidability of first-order logic (2 lectures)
- Axiomatic systems of number theory and Gödel's incompleteness theorems (both versions) (2 lectures)

3. **Introduction to Automata Theory on Infinite Inputs**

- Büchi and Rabin automata on infinite strings (2 lectures)
- Complementation and emptiness checking for Büchi automata, nondeterministic Büchi automata, Safra's theorem (3 lectures)
- Relation between decidability problems in logic and automata theory (2 lectures)
- Introduction to automata on tree inputs (2 lectures)

Evaluation:

- Midterm exam: 25% of total grade
- Final exam: 40% of total grade
- At least six sets of exercises: 25% of total grade
- Continuous assessment including several announced quizzes: 10% (max 5% extra credit)
- Research report and presentation (optional): up to 15% bonus points

References:

- G. Boolos, J. Burgess, and R. Jeffrey, *Computability and Logic*, 5th Edition, Cambridge University Press, 2007.
- D. Kozen, *Theory of Computation*, Springer, 2006.

- S. Hedman, *A First Course in Logic: An Introduction to Model Theory, Proof Theory, Computability, and Complexity*, Oxford University Press, 2004.
- M. Sipser, *Introduction to the Theory of Computation*, 2nd Edition, Thompson, 2006.

52 IT Strategic Planning and Management

Course Code: 40448, Units: 3, Prerequisite: IT Project Management, Co-requisite: None

Course Summary:

This course provides students with theoretical and practical knowledge of strategic IT management and planning within organizations. It covers selecting appropriate strategic study methods based on organizational capacity, using suitable frameworks, and producing transition solutions through adaptable engineering models. Students will understand the need for architectural roadmaps and their updates to move from current to desired states, enabling integration of solution systems crucial for national projects such as e-government. The course also develops students' skills to extract systemic solutions through practical exercises.

Course Outline:

- Introduction and course framework (2 sessions)
- Terminology of strategic management and planning
- Comprehensive 360-degree view of traditional strategic planning (4 sessions)
- Analytical tools: IFE, EFE, SPACE, SWOT, QSPM (2 sessions)
- Developing foundational organizational strategic plans (2 sessions)
- Types of strategic studies from business to technology (2 sessions)
- Organizational information architecture for managers (3 sessions)
- IT strategic planning (2 sessions)
- Hanschke's organizational architecture (2 sessions)
- Hanschke's IT landscape management (2 sessions)
- Hanschke's technical standards for organizational architecture (2 sessions)
- Reference models, policies, and change statements (1 session)
- Overview of organizational architecture methodologies (1 session)

- Königsberg's IT strategic planning (1 session)
- Organizational architecture in Iran: history and national model (1 session)
- Organizational architecture maturity models (1 session)
- From data governance to architecture governance to digital transformation including ITIL and COBIT (2 sessions)

Books Used:

- Inge Hanschke, *Strategic IT Management*, Springer, 2010.
- Danny Greefhorst and Erik Proper, *Architecture Principles*, Springer, 2011.
- Martin Op't Land, *Enterprise Architecture Creating Value by Informed Governance*, Springer, 2009.
- Mario Godinez, *The Art of Enterprise Information Architecture*, IBM Press, 2010.

Evaluation:

- Midterm and final exams: 12 points
- Quizzes: 6 points
- Individual study of latest technologies: 2 points

53 Computer Measurement and Control

Course Code: 40463, Units: 3, Prerequisite: Fundamentals of Electrical and Electronic Circuits, Co-requisite: None

Course Summary:

This course aims to familiarize students with various sensors and actuators, interface circuits, amplifiers, voltage level converters for sensor outputs and actuator commands, analog-to-digital and digital-to-analog converters, processor units, and other components of a computer-based (digital) control system.

Course Outline:

- Introduction to control processes (3 sessions)
  - Control systems, control process block diagrams, system evaluation, analog and digital processing, units, standards, and definitions, sensor response time, computational accuracy, and statistical quantities
- Analog signal shaping (4 sessions)
  - Basic principles, passive circuits, operational amplifier circuits
- Digital signal shaping (4 sessions)
  - Basic principles, converters, data acquisition systems
- Temperature sensors (4 sessions)
  - Metal resistors, thermistors, thermocouples, other temperature sensors
- Mechanical sensors (4 sessions)
  - Displacement, position and status sensors, force sensors, motion sensors, pressure sensors, fluid flow sensors
- Optical sensors (2 sessions)
  - Light intensity detectors, remote thermometry, light sources
- Final control elements (3 sessions)
  - Final control operations, signal conversion, industrial electronics, actuators, controller components

- Discrete state process control (2 sessions)
  - Definitions, system characteristics, relay controllers and ladder diagrams, programmable logic controllers
- Basics of controllers (1 session)
  - Process specifications, control system parameters, discontinuous, continuous, and combined control modes
- Analog controllers (1 session)
  - General capabilities, electronic controllers, pneumatic controllers
- Digital controllers (2 sessions)
  - Digital control methods, computer application in process control, digital data characteristics, controller software, examples of computer control

Books Used:

- Curtis D. Johnson, *Process Control Instrumentation Technology*, 7th Edition, Prentice-Hall International, Inc., 2006.
- Alan J. Crispin, *Programmable Logic Controllers and Their Engineering Applications*, McGraw-Hill, 1990.

Evaluation:

- Theoretical exercises: 4 points
- Midterm and final exams: 16 points

54 Information Technology

Course Code: 40467, Units: 3, Prerequisite: None, Co-requisite: None

Course Summary:

The broad domain of computer applications forms the framework of information technology (IT) discussions. This course introduces students to the principles, definitions, concepts, applications, organizational and social impacts, and managerial concepts of IT along with its foundations and architecture. Since computer and IT engineers are innovators and promoters of new solutions in this field, they must be aware of the latest concepts, achievements, and applications of IT globally and in Iran. The superficial breadth of concepts in this course provides a structural foundation for deeper study in subsequent courses.

Course Outline:

- Introduction (1 session)
  - First lecture, values and harms
  - Differences, similarities, and overlaps among Computer Science, Computer Engineering, Software Engineering, and Information Technology
  - Information (IT) and information systems in global standards
- History, definitions, principles, frameworks, and preconceptions (2 sessions)
  - From Wiener to Dreyfus, Toffler, Castells to Freeman's conceptualization
  - From Cybernetics to Computers, Informatics, and Information Technology
  - Impact perspective: technology is neither good nor bad but definitely not neutral (Kranberg)
- Data, information, and knowledge: definitions, differences, similarities, and IT values (3 sessions)
  - Definitions and relations of data, information, and knowledge
  - Shannon's information theory, Lucien Gérard's value of information
  - Cycles of data, information, and knowledge and their relationships

  - Types of information values
  - IT-based organizations in the digital economy and IT management

- Network computing and IT management in digital economy-based organizations (2 sessions)
  - Networks, tools, networking contracts, network types, internetworking, and the Internet
  - Evolution of automation in organizations
  - Telecommuting and virtual organizations

- IT absorption capacity, electronic readiness, digital rankings, criteria, and digital divide (2 sessions)
  - Absorptive capacity of technology, calculation and enhancement methods
  - Digital readiness and digital divide and the use of their measurements
  - Ranking models, parameters, calculation methods, and their values
  - Electronic readiness and its computational model

- E-commerce, business intelligence, and data warehouses (3 sessions)
  - Definitions, differences, and similarities between e-commerce and e-business
  - Various transactional relationships in e-commerce
  - Business models in the digital economy
  - Business intelligence: definitions, applications, and usage
  - Data warehouses: definitions, architecture, and their role in business intelligence
  - Types of data mining: data, text, web mining and applications in business intelligence

- Wireless, mobile, ubiquitous, pervasive, and value-adding computing (2 sessions)
  - Mobile and wireless communications: foundations and applications
  - Communication and information technologies and realization of ubiquitous computing
  - Pervasive computing and its requirements

  - Value-added computing, methods of realization and implementation requirements

- Local, enterprise, and international systems: features and integration (2 sessions)
  - Enterprise systems and priorities for their preparation
  - Global and international systems: design requirements and implementation features
  - Legacy systems: needs and integration solutions
  - Integration technologies and tools

- Management support systems, supply chains, enterprise resource planning, and customer relations (2 sessions)
  - Types of management information systems, strategic information, execution and decision support systems
  - Architecture and features
  - Applications and constraints

- Internet structures, IT foundations and architecture (3 sessions)
  - Intranets and extranets
  - Websites, blogs, social networks to enterprise portals and their types
  - Framework of an e-commerce architecture
  - Relationship between architecture and IT foundation in enterprises

- Contemporary value-added IT applications (2 sessions)
  - Geographic information systems: architecture and applications
  - Global positioning systems
  - Workflow management systems
  - Applications and utilization of remote sensing technology
  - Telecommuting, its facilities and consequences

- Effects, ethics, and security of IT (2 sessions)
  - Consequences of widespread IT presence globally
  - Necessity of IT ethics and manners, and their realization and implementation

  - Virtual world, second life, and social and cultural consequences
  - IT security and ways to achieve it

- Information society and e-government, electronic services and their foundations (2 sessions)
  - Definition of e-government: needs, requirements, and prerequisites for its realization
  - Information society: features and global realization requirements
  - E-learning: types, needs, and social impacts
  - Types and applications of electronic services

- National and international outlook of information technology (2 sessions)
  - History of IT in Iran
  - Stakeholders, laws, and governing documents of IT in Iran
  - IT industry and market in Iran
  - Electronic banking in Iran
  - Computer and IT education in Iran
  - Role of IT parks in technology transfer
  - Current global status of IT

Books Used:

- Linda Volonino and Efrain Turban, *Information Technology for Management: Improving Performance in The Digital Economy*, 8th Edition, WILEY, 2011.
- Efraim Turban, Dorothy Leidner, Ephraim McLean, and James Wetherbe, *Information Technology for Management: Transforming Organizations in the Digital Economy*, 5th Edition, John Wiley & Sons Inc., 2006.
- E. Turban, R. K. Rainer, and R. E. Potter, *Introduction to Information Technology*, 3rd Edition, WILEY, 2005.
- Urs Birchler and Monika Butler, *Information Economics*, Routledge, 2007.
- E. W. Martin and C. V. Brown, *Managing Information Technology*, 5th Edition, Prentice Hall, 2004.
- K. D. Willett, *Information Assurance Architecture*, CRC, 2008.

- Thomas H. Davenport and Laurence Prusak, *Information Ecology: Mastering the Information and Knowledge Environment*, OXFORD University Press, 1997.

Evaluation:

- Theoretical exercises: 7 points
- Midterm and final exams: 11 points
- Quizzes: 2 points

55 Agile Software Development

Course Code: 40475, Units: 3, Prerequisite: Systems Analysis and Design, Co-requisite: None

Course Objectives:
This course aims to familiarize undergraduate computer engineering students with advanced concepts, principles, and methods of Agile software system development. After reviewing Agile fundamentals and the XP methodology, students will learn about DSDM and DAD methodologies and use them alongside Agile patterns and practices to develop a software system.

Course Outline:

- Introduction: Overview of basic concepts and history of Agile development, Agile Manifesto and principles (1 session)
- XP (Extreme Programming) Methodology (2 sessions)
- DSDM (Dynamic Systems Development Method)
  - General framework, principles, and rules (2 sessions)
  - Feasibility Phase (1 session)
  - Foundations Phase (2 sessions)
  - Evolutionary Development Phase (2 sessions)
  - Deployment Phase (2 sessions)
  - Roles, deliverables, and Agile practices (3 sessions)
- DAD (Disciplined Agile Delivery)
  - General framework (1 session)
  - Inception Phase (1 session)
  - Elaboration Phase (2 sessions)
  - Construction Phase (2 sessions)
  - Transition Phase (1 session)
  - Iterative activities and Agile practices (2 sessions)
- Agile Practices: Team management, design, and Kanban (3 sessions)
- Patterns (3 sessions)

Evaluation:

- Exams: Midterm and final (60% of total grade)
- Exercises and Project: Exercises as part of a DSDM or DAD project, gradually completed and submitted during the semester (40% of total grade)

References:

- D. Wells, *Extreme Programming: A Gentle Introduction*, 2013. http://www.extremeprogramming.org
- DSDM Consortium, *The DSDM Project Framework Handbook*, Agile Business Consortium, 2014. https://www.agilebusiness.org/page/TheDSDMAgileProjectFram
- S. W. Ambler and M. Lines, *Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise*, IBM Press, 2012.
- Agile Alliance, *Agile 101: Subway Map to Agile Practices*, 2015. https://www.agilealliance.org/agile101/subway-map-to-agile-practices/
- E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.

56 Machine Learning

Course Code: 40477, Units: 3, Prerequisite: Artificial Intelligence, Linear Algebra, Co-requisite: None

Course Summary:

This course introduces the concepts of machine learning, providing familiarity with its various branches and important theoretical and practical aspects. Key techniques and algorithms across different branches are discussed. In supervised learning, regression and classification problems, their solutions, and model evaluation methods are covered. For classification, various perspectives and corresponding algorithms are introduced. Unsupervised learning topics include density estimation, unsupervised dimensionality reduction, and clustering. Finally, a brief introduction to reinforcement learning is provided.

Course Outline:

- Introduction to Machine Learning and Review of Probability and Linear Algebra (1 session)
- ML and MAP Estimation Methods (1 session)
- Regression (3 sessions)
  - Linear and Non-linear Regression
  - Overfitting
  - Error decomposition into Bias, Variance, and Noise
  - Regularization
  - Statistical Regression: Relation of SSE-based objective functions to ML and MAP estimates in regression
- Model Evaluation and Tuning (1-2 sessions)
  - Validation
  - Cross-validation
  - Model Selection
  - Feature Selection
- Classification

  - Probabilistic Classifiers (3 sessions)
    * Decision Theory and Bayes Optimal Classifier
    * Discriminative and Generative Probabilistic Classifiers
    * Binary and Multi-class Logistic Regression, Naïve Bayes
  - Classification using Discriminant Functions (6 sessions)
    * Perceptron
    * Fisher Linear Discriminant
    * Support Vector Machines (SVM) and Kernel Methods
    * Neural Networks
  - Decision Tree (1 session)
    * Information Gain and Entropy
    * ID-3 Algorithm
    * Tree Growth Stopping and Pruning
- Instance-Based Learning Methods (2 sessions)
  - Non-parametric Density Estimation
  - k-Nearest Neighbors Classifier
  - Locally Weighted Linear Regression
- Computational Learning Theory (2 sessions)
  - PAC Learning
  - VC Dimension
  - Structural Risk Minimization
- Ensemble Learning (2 sessions)
  - Boosting and Bagging
  - AdaBoost
- Unsupervised Dimensionality Reduction (2 sessions)
  - Principal Component Analysis (PCA)
  - Independent Component Analysis (ICA)
- Clustering (3 sessions)
  - Partitional Methods (k-means, EM + GMM)

  - Hierarchical Methods

- Reinforcement Learning (2 sessions)
  - Markov Decision Processes (MDP)
  - Model-based Learning Methods
  - Value Iteration and Policy Iteration
  - Model-free Learning Methods
  - SARSA, Q-learning, Temporal Difference Algorithms
- Advanced Topics in Machine Learning

Evaluation:

- Exercises: 20%
- Midterm: 25%
- Final Exam: 35%
- Quizzes: 10%
- Project: 10%

References:

- C. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
- T. Mitchell, *Machine Learning*, MIT Press, 1998.
- K. Murphy, *Machine Learning: A Probabilistic Perspective*, MIT Press, 2012.
- T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd Edition, 2008.

57 Application Engineering

Course Code: 40478, Units: 3, Prerequisite: None, Co-requisite: Systems Analysis and Design

Course Summary:

The main goal of this course is to connect students' knowledge from abstract IT subjects such as Strategic Management and E-Commerce with operational courses like Databases, Networks, and Programming. Secondary objectives include familiarity with components of IT solutions and methodologies for system creation by combining these components; understanding common systems and their application domains such as ERP, CRM, and Portals; acquaintance with middleware and platforms usable in designing IT solutions; concepts of modern system production and current technologies; and approaches to handling legacy systems in organizations. The targeted organizations are large, distributed entities requiring more complex and distributed IT solutions.

Course Outline:

- Introduction (3 sessions)
  - Organizational Strategies
  - Common Business Systems
  - Distributed Organizations and Systems
- Application Systems (7 sessions)
  - Definition of Application Systems
  - Common Application Systems such as ERP, CRM, Portal
  - Relationship between Application Systems and Organizational Strategies
  - Organizational Process Modeling
  - Identification of Application Systems based on Organizational Processes
  - Methodology for Identifying Application Systems
- Architecture (7 sessions)
  - Software Architecture

  - Data Architecture
  - Solution Architecture

- Systems Integration (8 sessions)
  - Approaches to Legacy Systems in Organizations
  - Integration of Systems with Each Other or with Legacy Systems
  - Data Warehousing and its Use for Integration
  - Strategies for Replacing or Renovating Legacy Systems
  - Reengineering Patterns

- Middleware and Modern Technologies for System Interaction (5 sessions)
  - Service-Oriented Architecture (SOA)
  - Web Services, CORBA, J2EE, etc.
  - Distributed Transaction Management
  - Asynchronous Message Exchange

Evaluation:

- Theoretical and Practical Exercises: 3 points
- Midterm and Final Exams: 15 points
- Quizzes: 2 points

References:

- Amjad Umar, *Enterprise Architectures and Integration with SOA – Concepts, Methodology and a Toolset*, NGE Solutions, 2010.
- Amjad Umar, *e-Business and Distributed Systems Handbook (from Strategies to Working Solutions)*, NGE Solutions, 2003.
- Hans-Erik Eriksson and Magnus Penker, *Business Modeling with UML*, 2000.

58 Hardware Description Languages

Course Code: 40483, Units: 3, Prerequisite: Digital Systems Design, Computer Architecture, Co-requisite: None

Course Summary:

This course aims to familiarize students with the required features of hardware description languages (HDLs) compared to software languages. It covers an overview and introduction to three well-known hardware design languages: VHDL, Verilog, and SystemC. Students will work with these languages and understand the important differences among them in hardware modeling. Additionally, the course explains the differences between hardware modeling and system modeling using SystemC.

Course Outline:

- SystemC Language and Hardware Modeling with It
  - History and evolution of SystemC
  - Modules and their components
  - Ports and their types, concept and applications of signals
  - Types of processes in SystemC and their uses
  - Data types in SystemC: two-valued logic, four-valued logic, computational data types, bitwise data types
  - Complex data types, defining and using structs for signals and ports
  - Implementation methods of combinational and sequential circuits with SystemC and differences between process types
  - State machines and types of Mealy and Moore machines, implicit and explicit state machine descriptions
  - Synthesis of SystemC models: combinational circuit synthesis, important notes to generate the desired circuit, preventing latch generation, sequential circuit synthesis and recommended style
  - Finite State Machine with Datapath (FSMD) model and its importance, implementation with SystemC
- VHDL Language and Hardware Modeling with It
  - History, evolution, and strengths of VHDL

  - Overview of language structure
  - Delay types in VHDL
  - Structural description, port connection methods, flip-flop design example, repetitive structure design examples
  - Parameterizing design and defining configurations
  - Data types, arrays, physical data
  - Multi-valued logic and related IEEE packages
  - Process statements and state machine design
  - Synthesizable subset and design styles
- Brief Review of Verilog and Qualitative Comparison of SystemC, VHDL, and Verilog

Evaluation:

- Theoretical Exercises: 3 points
- Midterm and Final Exams: 15 points
- Quizzes: 2 points

References:

- SystemC User's Guide, Version 2.0, SystemC Consortium, 2002.
- J. Bhaskar, *A SystemC Primer*, Star Galaxy Publishing, 2002.
- Peter J. Ashenden, *The Designer's Guide to VHDL*, Elsevier (Morgan Kaufmann), 2008.
- Z. Navabi, *VHDL: Analysis and Modeling of Digital Systems*, McGraw Hill, 1998.
- D. L. Perry, *VHDL: Programming by Examples*, McGraw Hill, 2002.

59 Object-Oriented Systems Design

Course Code: 40484, Units: 3, Prerequisite: Systems Analysis and Design, Co-requisite: None

Course Objectives:

This course introduces undergraduate software engineering students to the concepts, principles, and methods of object-oriented analysis and design of software systems. Students gain a thorough understanding of a modern (third-generation) object-oriented analysis and design methodology and become familiar with GoF design patterns and their practical application.

Course Content:

- Introduction - Overview of Object-Orientation and Evolutionary History of Object-Oriented Analysis and Design (1 session)
- Review of Unified Modeling Language (UML) (4 sessions)
- USDP Phases and Workflows
- The Four Phases (3 sessions)
- Requirements Workflow - Identifying and Modeling Use Cases (3 sessions)
- Analysis Workflow
  - Identifying and Modeling Analysis Objects and Classes (2 sessions)
  - Identifying and Modeling Relationships Between Analysis Objects and Classes (2 sessions)
  - Analysis Packages (1 session)
  - Realizing Use Cases in Analysis (2 sessions)
  - Activity Modeling (2 sessions)
- Design Workflow
  - Identifying and Modeling Design Objects and Classes (1 session)
  - Refining Relationships (1 session)
  - Interfaces and Components (1 session)
  - Realizing Use Cases in Design (1 session)

- Implementation Workflow (1 session)
- Design Patterns
  - Design Principles and Rules: The Six Fundamental Principles, GRASP Patterns, Contract-Based Design (1 session)
  - GoF Design Patterns
    * Creational Patterns: Factory Method, Abstract Factory, Builder, Prototype, Singleton (1 session)
    * Structural Patterns: Adapter, Bridge, Composite, Decorator, Facade, Proxy (1 session)
    * Behavioral Patterns: Chain of Responsibility, Iterator, Mediator, Memento, Observer, State, Strategy, Visitor (2 sessions)

Assessment:

- Exams: Midterm and Final Exams (60% of total grade)
- Exercises and Project: Exercises integrated into a course project on analysis and design, progressively developed and submitted during the semester (40% of total grade)

References:

- J. Arlow and I. Neustadt, *UML 2 and the Unified Process*, 2nd Edition, Addison-Wesley, 2005.
- H. Gomaa, *Software Modeling and Design: UML, Use Cases, Patterns, and Software Architectures*, Cambridge University Press, 2011.
- G. Booch, R. A. Maksimchuk, M. W. Engel, B. J. Young, J. Conallen, and K. A. Houston, *Object-Oriented Analysis and Design with Applications*, 3rd Edition, Addison-Wesley, 2007.
- E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.
- C. Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*, 3rd Edition, Prentice-Hall, 2004.

60 Introduction to Bioinformatics

Course Code: 40494, Units: 3, Prerequisite: Data Structures and Algorithms, Engineering Probability and Statistics, Co-requisite: None

Course Objectives:

This course aims to familiarize students with the essentials of bioinformatics data analysis. These essentials include a review of key topics in cellular and molecular biology, fundamental bioinformatics algorithms, statistical methods and machine learning techniques used in biomedical data analysis, bioinformatics databases, and practical data analysis using Linux OS and the R programming environment. Upon completion, students are expected to have the foundational knowledge needed to study advanced research and other courses in this field.

Course Content:

- Introduction
  - Necessity of Learning Bioinformatics
  - Applications of Bioinformatics in Biological and Medical Research
- Essentials of Cellular and Molecular Biology
  - Cell Components
  - DNA Structure and Replication
  - RNA Structure and Transcription
  - Protein Structure and Translation
  - Gene Expression Regulation
  - Cellular Differentiation
- Introduction to Bioinformatics Data
  - Technologies for Biological Data Generation: Microarray and Next Generation Sequencing
  - Important Biomedical Data Repositories
  - Integration of Various Databases
- Introduction to Statistical Methods

  - Gene Expression Difference Analysis
  - Statistical Tests
  - p-value and Its Correction Methods
  - Dimensionality Reduction in Biological Data
- Basic Biological Data Analysis Using R
  - Introduction to R Programming
  - Data Visualization in R
  - R/Bioconductor Libraries
  - Microarray Gene Expression Data Analysis
  - RNASeq Data Analysis
  - GO and Pathway Analysis
  - GSEA Analysis
  - Meta-Analysis of Data
- Introduction to Data Analysis on Linux Server
  - SSH Connection and Secure File Transfer
  - Bash Programming in Linux Environment
  - Direct Installation and Use of Bioinformatics Software
  - Introduction to BioConda
  - Parallel Execution of Software
- Introduction to Bioinformatics Algorithms
  - Biological Sequence Alignment
  - Phylogenetic Trees
  - Genome Assembly
  - Read Mapping (Alignment)
  - Motif Finding
- Systems Biology Analyses
  - Differential Equations Applications
  - Differentiation Analysis
  - Structural Data Analysis

- RNA and Protein Folding Problems
- Protein-Protein Interaction

Assessment:

- Workshop: 2 points
- Exercises: 5 points
- Project: 3 points
- Midterm Exam: 5 points
- Final Exam: 5 points

References:

- Bruce Alberts et al., *Essential Cell Biology*, Garland Science, 2013.
- Neil C. Jones and Pavel A. Pevzner, *An Introduction to Bioinformatics Algorithms*, The MIT Press, 2004.

61 Genetic and Evolution

Course Code: 40496, Units: 3, Prerequisite: None, Co-requisite: None

Course Objectives:

This course introduces students first to the principles of genetics at the cellular, molecular, and organismal levels, and then to the principles of genetics and evolution at larger scales, including populations and communities.

Course Content:

- Introduction to Genetics
- Genetics at the Cellular Level
  - Basics of Heredity
  - Revisiting and Extending Heredity Concepts
  - Genetic Linkage, Recombination, and Genetic Mapping
  - Genetics of Bacteria and Viruses
  - Chemical Identity of Genes
  - DNA Replication and Recombination
  - Chromosomal and Gene Mutations
- Introduction to Evolutionary Theory
  - Phylogenetic Nomenclature Code
  - Genotype-Phenotype Distinction
  - Definition of Life and Artificial Life
  - Population Genetics
  - Population and Unit of Natural Selection
  - Variation and Heredity
  - Fisher's Fundamental Theorem of Natural Selection
  - Fitness
  - DNA and RNA Molecules
  - Replication Fidelity and Mutation

– Genetic Drift in Finite Populations
– Criticism of Neo-Darwinism
– Central Dogma
– Age of the Universe
– Paleontological Evidence
– Lamarck and Weismann
– Observations from Domesticated Animals
– Elements of Evolution by Natural Selection
– Long-term Natural Selection Experiments
– Artificial Selection
– Main Evidence for Evolution
– Population Size and Growth Models
– Chaos Theory
– Evolution in Diploid Populations
– Diploid Organisms
– Hardy–Weinberg Principle
– Spread of a Beneficial Gene
– Dominant and Recessive Genes
– Types in Natural Populations
– Evidence for Genetic Variation
– Nature of Mutation
– Mutation-Selection Balance
– Mutation Load
– Genetic Recombination (Cross Over)
– Frequency-dependent Natural Selection
– Variable Environments
– Evolution at Multiple Loci
– Linkage Disequilibrium
– Mimicry in Butterflies
– Natural Selection and Linkage Disequilibrium
– Small and Structured Populations

- Population Structure
- Inbreeding
- Genetic Drift and Mutation
- Neutral Evolution and Molecular Evolution Rates
- Migration and Population Differentiation

References:

- Benjamin A. Pierce, *Genetics: A Conceptual Approach*, 5th Edition.
- Terry Brown, *Introduction to Genetics: A Molecular Approach.*
- John Maynard Smith, *Evolutionary Genetics*, OUP, 1999.
- Charlesworth, B. & Charlesworth, D., "Population Genetics from 1966 to 2016," 2016.
- Singh & Krimbas, *Evolutionary Genetics*, CUP, 1999.
- Charles W. Fox and Jason B. Wolf, *Evolutionary Genetics: Concepts and Case Studies*, Oxford University Press, 2006.
- Joseph Felsenstein, *Theoretical Evolutionary Genetics*, 2016.

62 Cellular and Molecular Biology

Course Code: 40495, Units: 3, Prerequisite: None, Co-requisite: None

Course Objectives:

This course introduces students to the fundamentals of cellular and molecular biology. It begins with the chemical bases of cellular biomolecules including proteins and nucleic acids, then discusses molecular processes such as DNA replication, transcription, and protein translation. Mechanisms controlling molecular processes in the cell are also covered.

Course Content:

- Chemical and Molecular Foundations of Biology
  - Molecules of Life, Genome, Cell Architecture and Function, Cell and Tissue
  - Chemical Foundations: Covalent bonds, Non-covalent interactions, Chemical components of the cell, Chemical reactions and chemical equilibrium, Bioenergetics
  - Protein Structure and Function: Hierarchical protein structure, Protein folding, Protein binding and enzyme catalysis, Regulation of protein function, Protein purification, tracing and characterization, Proteomics
- Genetics and Molecular Biology
  - Molecular basis of genetic mechanisms: Structure of nucleic acids, Transcription of protein-coding genes, Decoding of messenger RNA, Protein synthesis on ribosomes, DNA replication, DNA repair and recombination, Viruses
  - Molecular genetic methods: Genetic mutation analysis for gene study, DNA cloning, Cloning for gene expression studies, Identification of human diseases, Specific gene inactivation
  - Gene, Genome, and Chromosome: Eukaryotic gene structure, Chromosome organization and non-coding DNA, Mobile elements in genome, Genomics and gene expression
  - Transcriptional control of gene expression: Gene regulation in bacteria, Gene regulation in eukaryotes, Promoters, RNA polymerase II and transcription factors, Activation and repression mechanisms, Transcription factor activity control, Epigenetic transcription control

- Cell Structure and Function
  - Cell growth and observation, cell perturbation: Cell culture, Light microscopy and cell structure exploration, Protein localization in cells, Electron microscopy, Cell organelle isolation and study, Cell perturbation and functional studies

Assessment:

- Exercises and Projects: 30%
- Midterm and Final Exams: 70%

References:

- H. Lodish et al., *Molecular Cell Biology*, 9th Edition, W. H. Freeman, 2021.
- B. Alberts et al., *Molecular Biology of the Cell*, 7th Edition, W. W. Norton, 2022.

63 Microprocessor

Course Code: 40513, Units: 3, Prerequisite: Computer Architecture, Co-requisite: None

Course Summary:

This course builds on previous courses in computer structure and architecture, teaching students how to design, implement, program, evaluate, and debug a computer system using a microprocessor or microcomputer chip along with peripheral chips. It also introduces advanced microprocessors and how to utilize their advanced features.

Course Outline:

- Introduction to the evolution and development of microprocessors
- Introduction to a fundamental and widely-used processor
- Minimum mode:
  - 16-bit memory architecture and internal structure of 8086
  - Pin configuration and basic cycles
  - Instruction set and assembly programming
  - System architecture in minimum mode
  - Interrupts
- Maximum mode:
  - Pin configuration and basic cycles
  - System architecture in maximum mode
  - 8284 and 8288 chips
- Peripheral circuits:
  - Timers, serial and parallel ports
  - User interface (keyboard, display)
  - Environmental interfaces (rotary encoder, DA and AD converters)
  - Introduction to interrupt controller and DMA controller
- 8087 coprocessor:

  - Internal structure
  - Instruction set
  - Interaction with the main 8086 processor
- Introduction to a microcontroller
- Instruction set overview
- Interrupt management
- Peripheral circuits introduction:
  - Communication interfaces (I2C, SBI, etc.)
  - PWM unit
  - Various timers
  - Different ports
  - ADC and DAC units
- Power management unit introduction
- Introduction to a graphics processor
- Brief overview of PCI-Express interface
- Summary of graphics processor architecture and programming

Books Used:

- W. A. Triebel and A. Singh, *The 8088 and 8086 Microprocessors*, Prentice-Hall, 2003.
- M. A. Mazidi, *The 80x86 IBM PC & Compatible Computers*, Volume II, Prentice Hall International Inc., 1995.
- W. A. Smith, *ARM Microcontroller Interfacing: Hardware and Software*, Elektor International, 2010.
- M. A. Mazidi et al., *The AVR Microcontroller and Embedded Systems: Using Assembly and C*, Prentice Hall, 2011.
- D. Kirk and W. M. Hwu, *Programming Massively Parallel Processors*, Morgan Kaufmann, 2012.

Evaluation:

- Theoretical exercises: 3 points
- Mid-term and final exams: 15 points
- Quizzes: 2 points

64 Hardware Lab

Course Code: 40102, Units: 1, Prerequisite: Computer Architecture Lab, Co-requisite: None

Course Summary:

The aim of this laboratory course is to enhance students' skills in designing and implementing hardware systems for practical problems relevant nationally or globally. Applications include embedded systems, data acquisition and monitoring systems, Internet of Things, and digital systems in industrial and medical domains. Students are expected to apply their knowledge from hardware, computer architecture, operating systems, and system-level programming to build an efficient device for solving a real-world problem. The topics are project-based and may vary across semesters.

Course Outline:

- Implementation of a vital signs sampling and patient monitoring system using a mobile phone
- Design and simulation of a traffic control system
- Implementation of an access control system based on fingerprint/RFID card
- Design of vehicle positioning and parking spot status notification system in parking lots using cameras/proximity sensors/light sensors, etc.
- Visual matching detection of car body condition upon entry and exit using four cameras to check for damages during parking
- Implementation of at least one IoT-based or Cyber-Physical System application using sensors/actuators/modern boards
- Practical implementation of at least one hardware-in-the-loop experiment for signal processing or similar applications, preferably using Simulink/Matlab software

Books Used:

- M. A. Mazidi, Sh. Chen, and E. Ghaemi, *Atmel ARM Programming for Embedded Systems*, MicroDigitalEd, Vol 5, 2017.
- M. A. Mazidi, Se. Naimi, and Sa. Naimi, *AVR Microcontroller and Embedded Systems: Using Assembly and C*, MicroDigitalEd, 2017.

- R. H. Chu and D. D. Lu, "Project-based lab learning teaching for power electronics and drives," *IEEE Transactions on Education*, Vol. 51, No.1, pp. 108-113, 2008.

- J. Ma and J.V. Nickerson, "Hands-on, simulated, and remote laboratories: A comparative literature review," *ACM Computing Surveys*, Vol. 38, No. 3, 2006.