1.4 Computer Architecture

- **Course name:** Computer Architecture
- **Course number:** XXX

1.4.1 Course Characteristics

1.4.1.1 Key concepts of the class

- The fundamental principles for modern computer systems
- Computer instructions, their representation, and execution
- Computer arithmetics

1.4.1.2 What is the purpose of this course?

The course covers the fundamental principles of computer systems design. We first overview the key hardware components of a modern computer system, available performance metrics, and the general principles of computer architecture. We then discuss the representation and execution of computer instructions, instruction set architecture, the translation hierarchy of a high-level program into machine code. We also cover the elements of computer arithmetics, logic circuits, including combinational and sequential logic circuits. These theoretical principles are illustrated by using MIPS instruction set architecture, FPGA, and Verilog HDL programming language during the labs. We then study in details simple and pipelined implementation schemes of a processor, the idea of a pipelined execution, related hazards and their solutions. We complete the course by introducing several advanced topics, including computer security and vulnerabilities, GPU programming, and modern principles for memory hierarchy design.

1.4.1.3 Course objectives based on Bloom's taxonomy

- What should a student remember at the end of the course?

- Key components of a modern computer system
- Available performance metrics for computer systems
- Computer arithmetics operations, including floating point numbers
- Number systems and conversion between them
- Representation formats for computer instructions

- What should a student be able to understand at the end of the course?

- Fundamental principles of computer architecture (Moore's law, memory hierarchy, multiprocessing, speculative execution, and others)
- The design scheme of a modern processor
- The interaction principles between software and hardware
- Program representation and execution by a computer system

- What should a student be able to apply at the end of the course?

- The design skills of logic circuits by using Verilog HDL programming language
- FPGA programming by using Quartus Prime software
- MIPS assembly programming (including MARS simulator)

1.4.1.4 Course evaluation

Table 1.10: Course grade breakdown

Type	Default points	Proposed points
Labs/seminar classes	20	20
Interim performance assessment	30	20
Exams	50	60

1.4.1.5 Grades range

Table 1.11: Course grading range

Grade	Default range	Proposed range
A. Excellent	90-100	90-100
B. Good	75-89	70-89
C. Satisfactory	60-74	60-69
D. Poor	0-59	0-59

Each assignment will be assessed on a scale 0-10, where 6 is the minimum passing grade.

Labs/seminar classes points are divided into 5% Quizzes and 15% Labs deliverable. Quizzes will be conducted regularly during tutorials, and 3 worst quiz grades will not be considered for the final grade. Each lab is evaluated (0-2 points), where 0 is given for absence or lack of progress, 1 is given for approximately 50% of progress and 2 is given for perfect or almost perfect work on lab. All labs must be delivered before the exam; late lab delivery without reasonable justification will be penalized by 50%.

Exams will be conducted written as well as oral to assess the individual. Students failing the written part of the exam will not be invited for the oral part. The grading, though, is not a simple linear combination of the components above. In particular: Failing any part of the evaluation will trigger a failure in the entire course. If there are not failing components (all labs are delivered, all other course components, except the midterm, score at least 60%) the final grade will be computed as a weighted average of the components above approximated at the highest second digit and then rounded to the closest integer, and finally assigning A to 90, B to 70, and C to 60.

1.4.1.6 Resources and reference material

- D.A. Patterson and J.L. Hennessy. Computer Organization and Design: The Hardware/Software Interface. Morgan Kaufmann, 2014. ISBN 9780124077263
- W. Stallings. Computer Organization and Architecture: Design for Performance. Prentice Hall, 2010. ISBN 9780136073734
- Handouts supplied by the instructor
- Online resources shared by instructor

1.4.2 Course Sections

The main sections of the course and approximate hour distribution between them is as follows:

Table 1.12: Course Sections

Section	Section Title	Teaching Hours
1	Introduction to the Fundamental Concepts of Computer Architecture	18
2	Computational Logic Implementation in a Computer System	15
3	Instruction Representation and Execution in a Computer System	15
4	Computer Arithmetics	15
5	Processor Architecture	15
6	Advanced Topics	6

1.4.2.1 Section 1

Section title: Introduction to the Fundamental Concepts of Computer Architecture

Topics covered in this section:

- Key Components of a Computer System
- Fundamental Ideas of Computer Architecture
- Translation Hierarchy of a High-Level Program into Machine Code
- Performance Metrics of a Computer System

What forms of evaluation were used to test students' performance in this section?

Form	Yes/No
Development of individual parts of software product code	1
Homework and group projects	1
Midterm evaluation	1
Testing (written or computer based)	1
Reports	0
Essays	0
Oral polls	1
Discussions	1

Typical questions for ongoing performance evaluation within this section

Sample quistions from weekly quizzes:

1. Do you agree that main memory (RAM) is a non-volatile memory?
2. There are several types of memory available for computers, such as CPU cache, main memory (RAM), SSD, etc. What are the key differences between them?
3. What is the key principle behind the Von Neumann Architecture?
4. Specify a correct order for tools used during high-level program translation and execution: Compiler, Assebler, Linker, Loader;
5. Let a program run on a computer comprised of one processor only. Let us now increase the number of processors up to m>1, so that multiple instructions of that program can be executed in parallel. Assume that all processor speeds are the same. Do you agree that a program can never execute slower on m processors, as compared to the case of one processor?

Typical questions for seminar classes (labs) within this section

1. Demonstration and description of key elements of an FPGA board (memory unit, PCI slot, clock generator, etc.);
2. Description of specific features of FPGA as compared to other integrated circuit devices;
3. Writing basic code for FPGA board;
4. Configuration and usage of the basic functionality in Quartus Prime software

Test questions for final assessment in this section

1. Briefly describe the principles of Von Neumann architecture. Illustrate with a diagram.
2. Describe the steps that transform a program written in a high-level language such as C into a representation that is directly executed by a computer processor. Illustrate with a diagram; provide a brief description for each step.
3. Consider three different processors P1, P2, and P3 executing the same instruction set. P1 has a 3 GHz clock rate and a CPI of 1.5. P2 has a 2.5 GHz clock rate and a CPI of 1.0. P3 has a 4.0 GHz clock rate and has a CPI of 2.2. Answer the following questions: a) Which processor has the highest performance expressed in instructions per second? b) If the processors each execute a program in 10 seconds, find the number of cycles and the number of instructions. c) We are trying to reduce the execution time by 30% but this leads to an increase of 20% in the CPI. What clock rate should we have to get this time reduction?

1.4.2.2 Section 2

Section title: Computational Logic Implementation in a Computer System

Topics covered in this section:

- Logic Gates and Boolean Algebra
- Logic Circuits
- Combinational and Sequential Logic
- Number Systems
- The Basics of Verilog Hardware Description Language (HDL) Programming

What forms of evaluation were used to test students' performance in this section?

Form	Yes/No
Development of individual parts of software product code	1
Homework and group projects	1
Midterm evaluation	1
Testing (written or computer based)	1
Reports	0
Essays	0
Oral polls	1
Discussions	1

Typical questions for ongoing performance evaluation within this section

Sample questions from weekly quizzes:

1. Convert decimal number 123 into base-5 format;

2. Do you agree that a S/R latch and a D flip-flop have different storage capacities?
3. Choose the key differences between SRAM and DRAM memory types: cost, power consumption, volatility, access speed, storage capacity, etc.;
4. Do you agree that one of the key differences between sequential and combinational logic circuits is the presence of memory elements?

Typical questions for seminar classes (labs) within this section

1. Questions regarding the basic logic gates;
2. Assignments to design simple logic circuits with 2-3 logic gates on a white board;
3. Programming assignments in Quartus Prime software, to design and compile simple logic circuits;
4. Programming an FPGA board by using Verilog HDL in Quartus Prime environment, such as turning on or off leds based on a switch position;
5. Questions regarding the difference between combinational and sequential logic circuits;

Test questions for final assessment in this section

1. Prove that the AND and NOT logic gates can be implemented by using only the NOR logic gate.
2. What are the S/R latch and D latch? Draw the respective logic circuits. Describe the differences between them.
3. Briefly describe the key difference(s) between combinational and sequential logic circuits.
4. Define what a multiplexor logic circuit is (with an arbitrary number of inputs). Provide a truth table for a 2-to-1 multiplexor. Provide a logic circuit implementing a 2-to-1 multiplexor, that uses AND, NOT, and OR logic gates. Describe a Verilog module implementing such a logic circuit of a 2-to-1 multiplexor.

1.4.2.3 Section 3

Section title: Instruction Representation and Execution in a Computer System

Topics covered in this section:

- Instruction Set Architecture (ISA)
- The Overview of MIPS ISA

- Types of MIPS Instructions and Their Representation in a Binary Format
- Sample MIPS Assembly Programs

What forms of evaluation were used to test students' performance in this section?

Form	Yes/No
Development of individual parts of software product code	1
Homework and group projects	1
Midterm evaluation	1
Testing (written or computer based)	1
Reports	0
Essays	0
Oral polls	1
Discussions	1

Typical questions for ongoing performance evaluation within this section

Sample questions from weekly quizzes:

1. How many bits are in one MIPS word?
2. Which MIPS directive would you use to create a string data?
3. For MIPS instruction set architecture (ISA), each register is reserved for a specific purpose. Describe the purpose of registers listed below: $v0, $s0-$s7, $t0-$t7;
4. In MARS simulator for MIPS programming, all register values, that are displayed in the register viewer, start with prefix "0x". What is the meaning of this prefix?

Typical questions for seminar classes (labs) within this section Sample MIPS programming assignments in MARS simulator:

1. Print a "Hello, World!" message in a console;
2. Computation of a simple arithmetic expression for integer parameters;
3. Computation of the first 10 Fibonacci numbers;
4. Implementation of more advanced program structures, such as conditional loops

Test questions for final assessment in this section

1. Translate the following MIPS code to C (or pseudocode). Assume that variables f, g, h, and i are assigned to registers $s0, $s1, $s2, and $s3, respectively. Code to translate: sub $t0, $s1, $s2; addi $t0, $t0, 3; add $s0, $s3, $t0
2. Assume that two MIPS registers, $s0 and s1$, contain the following binary data (for simplicity, we assume 8-bit registers, rather that 32): $s0: 00100000; $s1: 01010101. What is the value of register $s1 after the execution of the following MIPS instruction?: sll $s1, $s0, 4.
3. List and describe the purpose of general-purpose MIPS registers.

1.4.2.4 Section 4

Section title: Computer Arithmetics

Topics covered in this section:

- Basic Arithmetic Operations (Bitwise, Shifts, Multiplication, Division, and Others)
- Overflow and Underflow Problems for Arithmetic Operations
- Arithmetic Operations with Floation Point Numbers
- Problems Related to Precision and Conversion for Floating Point Numbers

What forms of evaluation were used to test students' performance in this section?

Form	Yes/No
Development of individual parts of software product code	1
Homework and group projects	1
Midterm evaluation	0
Testing (written or computer based)	1
Reports	0
Essays	0
Oral polls	1
Discussions	1

Typical questions for ongoing performance evaluation within this section

Sample questions from weekly quizzes:

1. Assume that two MIPS registers, $s0 and $s1, contain the following binary data: $s0: 00100000; $s1: 01010101 (For simplicity, we assume 8-bit registers, rather that 32) What is the value of $s1 after the execution of the following instruction?: sll $s1, $s0, 4
2. What is a "register spilling" in the context of MIPS instruction set architecture?
3. Do you agree with the following statement? In some cases, MIPS logical shift operations, sll and srl, can be used as an efficient alternative to multiplication and division operations, mul and div.
4. Do you agree that overflow and underflow exceptions correspond to cases, when the result of an arithmetic operation surpasses and subceeds, respectively, the maximum and the minimum value for an appropriate data type returned by that arithmetic operation?

Typical questions for seminar classes (labs) within this section Sample MIPS programming assignments in MARS simulator, to practice floating-point operations:

1. Division of two floating-point numbers;
2. Conversion of Fahrenheit into Celsius temperature, and vice versa;

3. Computation of a sphere surphase area;
4. Questions regarding the execution of arithmetic operations with interger and floating-point values

Test questions for final assessment in this section

1. Briefly describe the overflow and underflow problems for arithmetic operations.
2. Describe the difference between executing arithmetic operations with integers and floating-point values for a MIPS processor.
3. What is a precision problem for a floating-point operation?

1.4.2.5 Section 5

Section title: Processor Architecture

Topics covered in this section:

- Key Components of a Processor: Control and Arithmetic Logic Unit, Registers
- Processor Datapath and Control Signals
- The Notion of a Pipelined Execution, Pipeline Hazards, and Their Solutions
- A Simple and Pipelined Implementation Schemes of a Processor

What forms of evaluation were used to test students' performance in this section?

Form	Yes/No
Development of individual parts of software product code	1
Homework and group projects	1
Midterm evaluation	0
Testing (written or computer based)	1
Reports	0
Essays	0
Oral polls	1
Discussions	1

Typical questions for ongoing performance evaluation within this section

Sample questions from weekly quizzes:

1. Do you agree that the key motivation for the CPU pipelining is to speed-up the execution of a program by exploring multiple CPU cores?
2. Which CPU block(s) is/are accessed during the execution of the following instruction? lw $1, 5($2)
3. What are 5 major stages of a pipelined instruction execution?

4. Do you agree that, for a processor with 5 pipelined stages, the number of concurrently executed instructions is up to 4?
5. There are several types of processors available, including single-cycle and multicycle.The major advantage of a single-cycle processor is the simplicity of its design. But what is its key drawback?

Typical questions for seminar classes (labs) within this section

1. Design of a testbench in ModelSim for Quartus Prime programming environment;
2. The design of Half-Adder, Full-Adder, Ripple Carry Adder by using Verilog HDL in Quartus Prime
3. Testing the correctness of Verilog HDL design by using ModelSim

Test questions for final assessment in this section

1. What is a Program Counter (PC) register of a processor?
2. Describe the principle of a pipelined CPU execution. Provide a diagram illustrating the concept. Briefly describe the 5 key stages of a classical pipeline.
3. What are the key differences between Control Unit (CU) and Arithmetic Logic Unit (ALU) of a processor? Which purposes do they serve?
4. What is a CPU datapath?

1.4.2.6 Section 6

Section title: Advanced Topics

Topics covered in this section:

- Computer Security and Vulnerabilities
- Graphics Processing Unit (GPU) and General-Purpose GPU Programming
- Modern Approaches for Memory Hierarchy Design

Typical questions for ongoing performance evaluation within this section
What forms of evaluation were used to test students' performance in this section?

1. Cold boot attack explores vulnerabilities in a memory dump mechanism. What is a memory dump?
2. Below is a list of possible vulnerability attacks. Choose the one(s) that explore(s) vulnerabilities in a speculative execution of modern processors: Meltdown, Foreshadow, Cold boot attack, Spectre, No choice is correct;

Form	Yes/No
Development of individual parts of software product code	1
Homework and group projects	1
Midterm evaluation	0
Testing (written or computer based)	1
Reports	0
Essays	0
Oral polls	1
Discussions	1

3. Choose the most precise definition for a side-channel attack: An attack that explores vulnerabilities in the hardware implementation of a computer system, An attack that explores vulnerabilities in the software components of a computer system;
4. Do you agree that Meltdown and Spectre vulnerabilities both explore race conditions in existing memory circuits?

Typical questions for seminar classes (labs) within this section

1. Programming assignment to implement Multiplexor using Verilog HDL in Quartus Prime;
2. Performance optimization of a Verilog HDL design;
3. The design of a simple Arithmetic-Logic Unit (ALU);
4. Revision questions

Test questions for final assessment in this section

1. What is an out-of-order execution? What hardware features of CPU implementation, in addition to an out-of-order execution, are exploited by Meltdown vulnerability? How serious is Meltdown vulnerability?
2. What is an instruction-level parallelism?
3. Describe the idea of a general-purpose GPU programming.
4. Briefly explain the working principles of a CPU cache.
5. Discuss advantages and drawbacks of a hierarchical memory model for computer systems.