<table>
<tr><td rowspan="7">Identification</td><td>Subject</td><td>CMS 240: Computer Organization - 6ECTS</td></tr>
<tr><td>Department</td><td>Computer Science</td></tr>
<tr><td>Program</td><td>Undergraduate</td></tr>
<tr><td>Term</td><td>Spring, 2023</td></tr>
<tr><td>Instructor</td><td>Azeem Akram</td></tr>
<tr><td>E-mail:</td><td>hm.azeem.akram@outlook.com</td></tr>
<tr><td>Classroom/hours</td><td></td></tr>
<tr><td>Prerequisites</td><td colspan="2">English</td></tr>
<tr><td>Language</td><td colspan="2">English</td></tr>
<tr><td>Compulsory/Elective</td><td colspan="2">Required</td></tr>
<tr><td>Required textbooks and course materials</td><td colspan="2">Core textbooks:

1. Digital Design and Computer Architecture, David Harris, Sarah L. Harris, 2nd ed., Morgan Kaufmann.
2. Computer organization and design, D. A. Patterson and J. L. Hennessy, 5th ed., Morgan Kaufmann.</td></tr>
<tr><td>Course outline</td><td colspan="2">Overview of computer technologies, Computer architecture, instruction set architecture (ISA), ISA design considerations, RISC vs. CISC, assembly and machine language, translation and program start-up. Computer arithmetic, arithmetic logic unit, floating-point numbers and their arithmetic implementations. Processor design, data path and control implementation, pipelining, hazards, pipelined processor design, hazard detection and forwarding, branch prediction and exception handling. Memory hierarchy, principles, structure, and performance of caches, virtual memory.</td></tr>
<tr><td>Course objectives

Learning outcomes</td><td colspan="2">The objective of a "Computer Organization" course is to provide students with a solid understanding of the structure and function of computer systems at the machine level, including the fundamental concepts of computer architecture and assembly language programming. This course typically covers topics such as:
• Computer architecture and components, such as the CPU, memory, and I/O devices.
• Assembly language programming, including MIPS instruction set architecture, memory addressing, and data manipulation.
• System-level programming, including interrupt handling, input/output operations, and interrupt-driven I/O.
• Low-level memory management and optimization techniques.
The ultimate goal is to enable students to develop a deep understanding of how computers work, and to equip them with the skills necessary to write efficient and effective low-level code.

The learning outcomes are as follows:
1. Understanding of computer architecture: Students will be able to understand the components of a computer system, and the relationships between these components.
2. Assembly language programming skills: Students will be able to write assembly language programs, using a specific instruction set architecture, and will have a solid understanding of memory addressing and data manipulation.
3. System-level programming: Students will have the ability to write system-level programs, including interrupt handling and input/output operations.
4. Low-level memory management: Students will understand how memory is managed at the low-level, including the use of stacks, heaps, and other memory management techniques.
5. Optimization techniques: Students will understand how to optimize code for performance, including the use of different memory access patterns and data structures.
Overall, the course aims to provide students with a comprehensive understanding of computer systems and the skills necessary to write low-level code for various applications.
.</td></tr>
</table>

Teaching methods	**Lecture**		x
	Group discussion		x
	Experiential exercise		x
	Lab		x
	Case analysis		
	Course paper		x
	Others		
Evaluation	**Methods**	**Date/deadlines**	**Percentage (%)**
	Midterm Exam		30
	Case studies		
	Class Participation		10
	Quizzes		20
	project		
	Presentation/Group Discussion		
	Final Exam		40
	Others		
	Total		100
Policy	**Preparation for class** The lecture material will focus on the major points introduced in the text. Reading the assigned chapters and having some familiarity with them before classwill greatly assist your understanding of the lecture. After the lecture, you should study your notes and work relevant problems. **Quizzes**: Three in-class quizzes will be given; the highest two will be considered.		

- **Withdrawal (pass/fail)**
  This course strictly follows grading policy of the School of Engineering and Applied Science. Thus, a student is normally expected to achieve a mark of at least 60% to pass. In case of failure, he/she will be required to repeat the course the following term or year.

- **Cheating/plagiarism**
  Cheating or other plagiarism during the Quizzes, Mid-term and Final Examinations will lead to paper cancellation. In this case, the student will automatically get zero (0), without any considerations.

- **Professional behavior guidelines**
  The students shall behave in the way to create favorable academic and professional environment during the class hours. Unauthorized discussions and unethical behavior are strictly prohibited.

- **Ethics**
  Students should not arrive in late to class.
  All cell phones must be turned off and stowed away before entering class.
  Use of any electronic devices is not allowed in the classroom and violators will be punished accordingly.

WK	Date/Day (tentative)	Topics	Textbook/Assignments
1		**Introduction to Computer Organization and Architecture** • The von Neumann Architecture • System Bus • Clock • Memory • Arithmetic Logic Unit • Registers • Control Unit • Input Output Devices • Interrupts	Lecture Slides
2		**Computer Architecture** • Instructions • Program Counter • Fetch-Decode-Execute Cycle • Instruction Fetch • Instruction Decode • Data Fetch • Instruction Execution • Result Return	Lecture Slides
3		**Computer Architecture** • Memory • Primary storage • Registers • Cache • Random Access Memory (RAM) • Read Only Memory (ROM) • Secondary storage • Magnetic • Optical	Lecture Slides
4		**Assembly Language** • Instructions • Registers set • Memory • Constants/immediate **Machine Language** • R-Type Instructions • I-Type Instructions • J-Type Instructions • Interpreting Machine Language Code • The Power of the Stored Program	**Quiz#01** Lecture Slides Classroom Practice
5		**Assembly Programming** • Arithmetic/Logical Instructions • Shift Instructions • Multiplication and Division Instructions • Conditional Branches	Lecture Slides Classroom Practice
6		**Assembly Programming** • Jumps Conditional Statements • If Statements • If/Else Statements • Switch/Case Statements.	Lecture Slides Classroom Practice

7		**Assembly Programming** Getting Loopy • While Loops • For Loops • Magnitude Comparison • Arrays • Function Calls	**Quiz#02** Lecture Slides Classroom Practice
8		**Midterm Exam**	
9		**Assembly Programming** • Addressing Modes • Compiling • Assembling • Loading • Pseudo instructions • Exceptions • Signed and Unsigned Instructions • Floating-Point Instructions.	Lecture Slides Classroom Practice
10		**MIPS microprocessor** • Architectural • Instruction Set • Design Process • MIPS Microarchitectures • Performance Analysis	Lecture Slides
11		**SINGLE-CYCLE PROCESSOR** • Single-Cycle Datapath • Single-Cycle Control • Performance Analysis	Lecture Slides
12		**PIPELINED PROCESSOR** • Pipelined Datapath • Pipelined Control • Hazards • Performance Analysis	**Quiz#03** Lecture Slides
13		• Memory System Performance Analysis • Caches • Advanced Cache Design, • The Evolution of MIPS Caches	Lecture Slides Classroom Practice
14		**Scheduling** Scheduling Algorithms • FCFC, RR, SJF, PQ	Lecture Slides Classroom Practice
15		**Scheduling** Algorithms Performance Analysis • FCFC, RR, SJF, PQ	Lecture Slides Classroom Practice
	TBA	**Final Exam**	

This syllabus is a guide for the course and any modifications to it will be announced in advance.