AIN SHAMS UNIVERSITY
I-Credit Hours Engineering Programs
(i.CHEP)

Research Project

Program on which the course is given	**Computer Engineering and Software Systems**
Course Code	CSE112
Course Title	Computer Organization and Architecture
Instructor	Dr. Tamer Mostafa
Deadline	3/6/2020
ILOs examined	**Knowledge** 1. Explain the MIPS instruction set architecture 2. Explain how computers execute programs 3. Illustrate how a data-path with or without pipelining is designed. 4. Infer the effect of the memory hierarchy on the performance **Thinking skills** 5. Write MIPS assembly programs 6. Translate C programs to assembly and assembly programs to machine code **Subject-based practical skills** 7. Design a processor's data path **Skills for life and work (general skills)** 8. Refer to relevant literature search for information and engage in life-long self-learning discipline. 9. Develop problem solving and creative thinking 10. Work and communicate effectively in team by effective collaboration and task management, working in a constrained stressful environment, and leading and motivating individuals.

The project consists of two parts:

Part-A) Designing and simulating a mini MIPS processor that should execute a specific code (40%)

Part-B) Designing and implementing an assembler and emulator that converts MIPS instructions into machine code and emulates its execution. (60%)

The project (two parts) is to be done in teams, each of a maximum of 5 students. The par of each student should be explicitly written in the report.

Part-A marking criteria

Course Code:	**CSE112**	Course Name:	**Computer Organization and Architecture**	**Weight**	40%	Deadline	**3/6/2020**
Student Name:	....	Student ID:	....				

	A (89-100)				B (76-88)				C (67-75)				D (60-66)				F (0-59)			
	100	96	92	89	88	84	80	76	75	72	69	67	66	64	62	60	59	40	20	0
Completeness and creativity (30%)	• Design is complete and give all the required outputs with the best performance				• Design is complete but does not give all the required outputs				• Design is complete but gives wrong outputs				• Design is partially complete				• Design is wrong or missing main components			
Simulation and results (40%)	• Simulation is complete and works efficiently				• Simulation is complete but does not give all the required outputs				• Simulation is complete but gives wrong outputs				• Simulation is partially complete				• Simulation is completely wrong or missing main components			
Report (30%)	• Excellent ability to express ideas with proper language and technical vocabulary.				• Good ability to express ideas with proper language and technical vocabulary.				• Normal ability to express ideas with proper language and technical vocabulary.				• Low ability to express ideas with proper language and technical vocabulary.				• Difficult to express ideas with proper language and technical vocabulary.			

1st marker Total	....	1st marker Signature	....	**ASU** Agreed Mark	....
2nd Marker Total	....	2nd marker Signature	....		

General Comments:

....

....

....

....

....

....

....

....

UEL Grading System	**Agreed Mark**	**ASU Grading Scale**	
% equivalent at UEL	**Range**	**% at ASU**	**Grade**
95% and higher		97% and higher	A+
82% to less than 95%		93% to less than 97%	A
70% to less than 82%		89% to less than 93%	A-
66% to less than 70%		84% to less than 89%	B+
63% to less than 66%		80% to less than 84%	B
60% to less than 63%		76% to less than 80%	B-
56% to less than 60%		73% to less than 76%	C+
53% to less than 56%		70% to less than 73%	C
50% to less than 53%		67% to less than 70%	C-
45% to less than 50%		64% to less than 67%	D+
40% to less than 45%		60% to less than 64%	D
Less than 40%		Less than 60%	F

1. Part-A: Designing and simulating a mini MIPS processor that should execute a specific code

Requirements:

- Design and implement a program that shows the execution of the following specific code on a pipelined MIPS processor.

```
    slti $at, $s5, 5
    beq $at, $zero, Else
    add $s6, $s5, $zero
    j Exit
Else: add $s6, $zero, $zero
Exit:
```

- Use one of the following programming languages: any of the C family, Java, python.
- Provide screen shots of the emulated execution.
- The program should enable the user to trace the code execution line by line.
- The program should show the active stage of each instruction at each clock cycle.
- The input file could be a text file that contains the MIPS code.

Delivery procedure:

- Two files should be delivered in a folder named "CSE112_PA_G#", where G# is the group number:
  - PA_G#_report: The documentation of the project that follows the implementation procedure. After the title page, you should include a page declaring the role of each student in the project.
  - PA_G#_code:
    - Source code of the program; fully documented.
- All files to be submitted in (.DOC, .DOCX) format ONLY.
- Project report to be Grammarly checked by student.
- Use at least 3 references and, at least, one of them must be an updated reference (after 2010). (for the two parts)
- Project report might undergo similarity checks for allocating plagiarism/cheating cases.

Part-B marking criteria

Course Code:	**CSE112**	Course Name:	**Computer Organization and Architecture**	**Weight**	60%	Deadline	**3/6/2020**		
Student Name:	....					Student ID:	....		

	A (89-100)				B (76-88)				C (67-75)				D (60-66)				F (0-59)			
	100	96	92	89	88	84	80	76	75	72	69	67	66	64	62	60	59	40	20	0
Completeness and creativity (30%)	• Design is complete and give all the required outputs with the best performance				• Design is complete but does not give all the required outputs				• Design is complete but gives wrong outputs				• Design is partially complete				• Design is wrong or missing main components			
Implementation and results (40%)	• Implementation is complete and works efficiently				• Implementation is complete but does not give all the required outputs				• Implementation is complete but gives wrong outputs				• Implementation is partially complete				• Implementation is completely wrong or missing main components			
Report (30%)	• Excellent ability to express ideas with proper language and technical vocabulary.				• Good ability to express ideas with proper language and technical vocabulary.				• Normal ability to express ideas with proper language and technical vocabulary.				• Low ability to express ideas with proper language and technical vocabulary.				• Difficult to express ideas with proper language and technical vocabulary.			

1st marker Total	....	1st marker Signature	....	**ASU** Agreed Mark	....
2nd Marker Total	....	2nd marker Signature	....		

General Comments:

....

....

....

....

....

....

....

....

UEL Grading System	**Agreed Mark**	**ASU Grading Scale**	
% equivalent at UEL	**Range**	**% at ASU**	**Grade**
95% and higher		97% and higher	A+
82% to less than 95%		93% to less than 97%	A
70% to less than 82%		89% to less than 93%	A-
66% to less than 70%		84% to less than 89%	B+
63% to less than 66%		80% to less than 84%	B
60% to less than 63%		76% to less than 80%	B-
56% to less than 60%		73% to less than 76%	C+
53% to less than 56%		70% to less than 73%	C
50% to less than 53%		67% to less than 70%	C-
45% to less than 50%		64% to less than 67%	D+
40% to less than 45%		60% to less than 64%	D
Less than 40%		Less than 60%	F

Part-B: MIPS Simulator and Emulator

1- Requirements:

- Design and implement a program that converts MIPS assembly instructions into machine code, and emulates its execution.
- Use one of the following programming languages: any of the C family, Java, python.
- Provide screen shots of the emulated execution.
- The program should enable the user to trace the code execution line by line.
- A minimum of 19 instructions should be implemented. The instruction set should include:
    - Arithmetic operations: ADD, ADDI, SUB,
    - Logic operations: AND, ANDI, OR, ORI, NOR, SLL, SRL, SRA
    - Compare instructions: STL, SLTI
    - Move instructions: LUI
    - Load and store: LW, SW
    - Branch instruction: BEQ, BNE, J
- The input file could be a text file that contains the MIPS code.
- As a test to the program, you could show the implementation of the following code

```
    slti $at, $s5, 5
    beq $at, $zero, Else
    add $s6, $s5, $zero
    j Exit
Else: add $s6, $zero, $zero
Exit:
```

- You do not need to write a full program to be executed.
- You can assume the variables as memory locations that are pre-allocated.

2- Delivery procedure:

- Two files should be delivered in a folder named "CSE112_PB_G#", where G# is the group number:
    - PB_G#_report:
        - The documentation of the project that follows the implementation procedure.
        - Screen shots of the execution
        - After the title page, you should include a page declaring the role of each student in the project.
    - PB_G#_code:
        - Source code of the program; fully documented.

- All files to be submitted in (.DOC, .DOCX) format ONLY.
- Project report to be Grammarly checked by student.
- Use at least 3 references and, at least, one of them must be an updated reference (after 2010). (for the two parts)
- Project report might undergo similarity checks for allocating plagiarism/cheating cases.