The University of Lahore

(Computer Organization and Assembly Language)

Lab Task-06

Issue Date: 16-04-2025   Points 10

Due Date: 16-04-2025 (11:59pm)

Lab Instructor: Mr. Muhammad Hassan

Important points to be observed for assignment submission.

- **Write your own solution; copied work will result in mark deductions.**
- **25% marks will be deducted for each day of late submission.**
- **Submit the task in soft form.**
- **Using AI or cheating will result in a zero grade.**
- **No late submissions will be accepted**

Task Submission Process:

- Ensure your document is neat, properly formatted, and follows the instructions.
- Submit the task in Hard Form on Thursday
- Any plagiarism will result in a zero grade.
- Upload the WORD file on Slate.

Learning Outcomes:

- Accept user input through interrupts.

- Store and retrieve data using stack instructions (PUSH, POP).

- Perform conditional checks using CMP and jump instructions.

- Use loops to automate repetitive checks in Assembly language.

Scenario Title:

User-Defined Student Marks Evaluation Using Assembly Logic

Scenario:

You are part of a college's microprocessor lab development team, assigned to build a lightweight evaluation utility in Assembly Language for the internal assessment system. This tool is designed to help the examination department check whether students have passed or failed a subject based on their marks.

The system must be capable of processing marks for three students, where:

1) Each student manually enters their marks via keyboard input.
2) The system should push each mark onto the stack, simulating how the processor temporarily saves data before processing.
3) Using a loop, the program should:
   - Pop each mark from the stack.
   - Compare it with a fixed passing threshold (e.g., 50).
   - Display either:
     - "Pass" → if the mark is greater than or equal to 50, or
     - "Fail" → if the mark is less than 50.

This task combines key concepts from low-level programming including stack memory, looping structures, and conditional logic using CMP and jump instructions.