****

**UNITED STATES INTERNATIONAL UNIVERSITY AFRICA  
NAIROBI, KENYA  
APT2022B: INTRODUCTION TO ASSEMBLY PROGRAMMING**

**CAT**

**SPRING SEMESTER 2024**

**INSTRUCTOR: LINUS ALOO**

**DATE: 18TH MARCH, 2024 TIME: 45MINS**

**INSTRUCTIONS**

1. Answer **ALL** Questions.
2. Write your **name** and **admission number**.

**ADM.NO……668201………………NAME……Mohameddeq Abdi….…………………**

a) Fig. Q1a) shows the structure of a section of the stack memory for 8086 microprocessor use.

**[8marks]**



Fig. Q1a)

1. Explain the working principle of the stack memory.

* **The stack memory in the 8086 microprocessor is a region of memory used for storing data temporarily during program execution.**
* **The stack pointer (SP) register keeps track of the top of the stack. When data is pushed onto the stack, the stack pointer is decremented, and when data is popped off the stack, the stack pointer is incremented.**
* **It operates on a Last-In-First-Out (LIFO) basis, meaning the last data pushed onto the stack is the first to be popped off.**
* **The stack is commonly used for storing return addresses, local variables, and passing parameters to subroutines**.

1. Given PUSH and POP instructions, match them with arrows A and B.

* **Arrow A corresponds to the PUSH instruction, which pushes data onto the stack.**
* **Arrow B corresponds to the POP instruction, which pops data off the stack.**

1. Write assembly language program statements to retrieve the data 1125H from the stack memory and store in AX.

**POP AX ; Retrieve the data from the stack and store it in AX**

1. Assuming AX=3344H, write assembly language program statements to store the data in AX into the stack memory from location 04H.

**PUSH AX ; Store the data in AX onto the stack**

b) Write 8086 ALP to perform the following: **[8marks]**

1. Store the data 3039H and 10F9H into memory locations 2000H and 2002H resp.
2. Load the contents of memory locations in (1) into AX and BX regs. resp
3. Compare the two numbers
4. If equal, perform signed multiplication on the numbers and then negate the answer(s)
5. If not equal, find their difference and perform both ASCII and BCD corrections on the answer
6. Store the answer(s) into memory locations starting from address 2006H
7. End the program.
8. Emulate the program in emu 8086 and provide a screen shot of key areas (Report Window, Machine code, Accumulator and Flags) with explanations

**; Store the data 3039H and 10F9H into memory locations 2000H and 2002H respectively**

**MOV WORD PTR [2000H], 3039H ; Store 3039H into memory location 2000H**

**MOV WORD PTR [2002H], 10F9H ; Store 10F9H into memory location 2002H**

**; Load the contents of memory locations into AX and BX registers respectively**

**MOV AX, [2000H] ; Load the contents of memory location 2000H into AX**

**MOV BX, [2002H] ; Load the contents of memory location 2002H into BX**

**; Compare the two numbers**

**CMP AX, BX ; Compare the two numbers**

**; If equal, jump to the multiplication section**

**JE multiplication**

**; If not equal, jump to the difference section**

**JNE difference**

**multiplication:**

**; Perform signed multiplication on the numbers**

**IMUL BX ; Perform signed multiplication on the numbers**

**NEG AX ; Negate the answer**

**; Jump to store the result**

**JMP store\_result**

**difference:**

**; Find the difference**

**SUB AX, BX ; Find the difference**

**; Perform ASCII correction**

**AAD ; Perform ASCII correction**

**; Perform BCD correction**

**DAA ; Perform BCD correction**

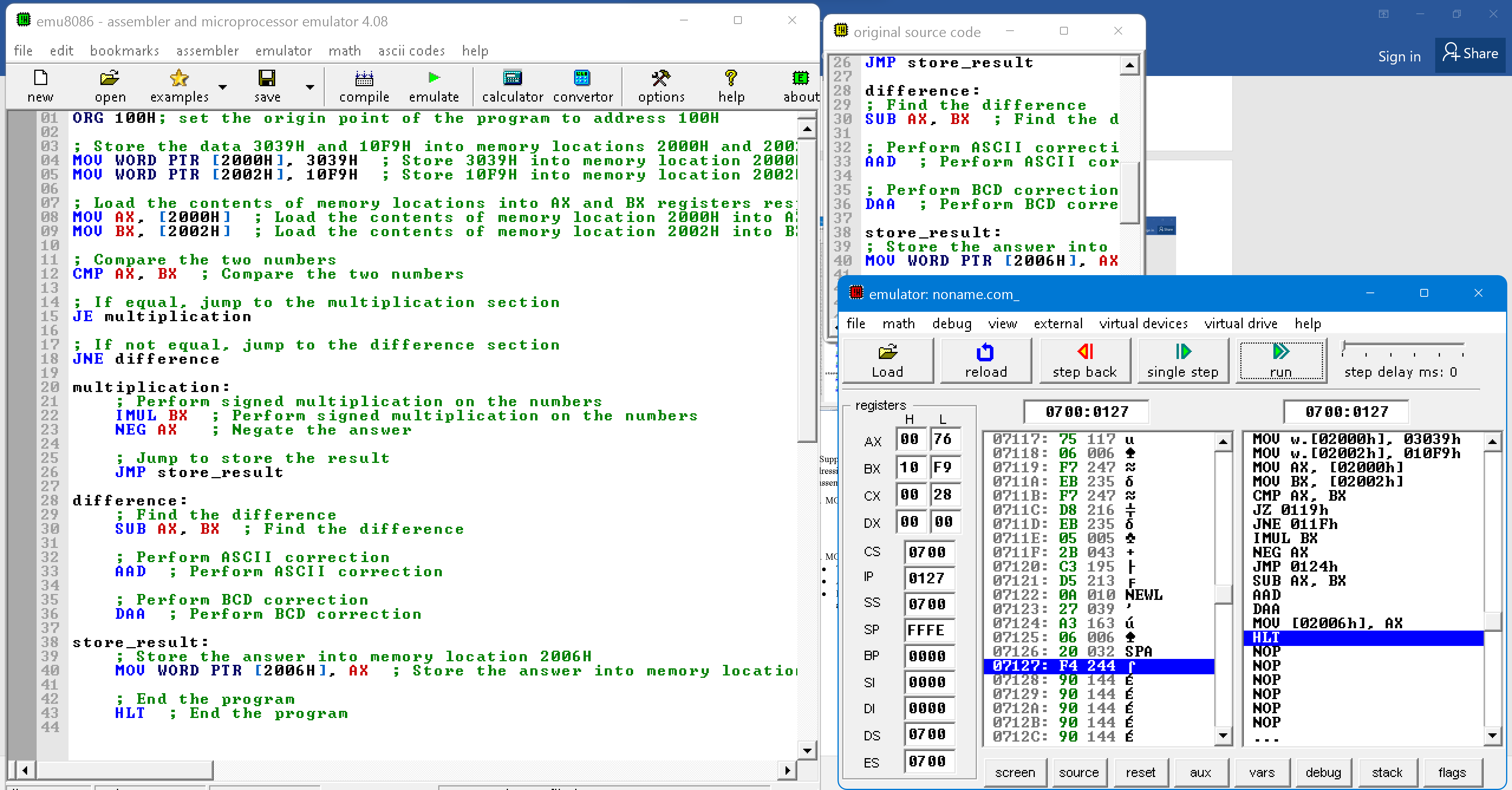
**store\_result:**

**; Store the answer into memory location 2006H**

**MOV WORD PTR [2006H], AX ; Store the answer into memory location 2006H**

**; End the program**

**HLT ; End the program**

****

c) Suppose that DS=5300H, BX=0300H, LIST=0450H, and SI=0500H. Identify the type of addressing mode involved and compute the address accessed by each of the following instructions in assembly programming and also give their meanings: **[4marks]**

1. MOV LIST[SI], DX

* **Type of addressing mode: Base/Index with Displacement**

**Effective address = LIST + SI = 0450H + 0500H = 0950H**

**Physical address = (DS \* 16) + Effective address = (5300H \* 16) + 0950H = 85250H**

**Address accessed: 85250H**

* **Meaning: Move the contents of register DX into the memory location pointed to by the sum of LIST and SI.**

1. MOV CX, [BX+SI]

* **Type of addressing mode: Base/Index with Displacement**

**Effective address = BX + SI = 0300H + 0500H = 0800H**

**Physical address = (DS \* 16) + Effective address = (5300H \* 16) + 0800H = 85000H**

**Address accessed: 85000H**

* **Meaning: Move the contents of the memory location pointed to by the sum of BX and SI into register CX**.

**END**