

Experiment 9

Aim: Find the factorial of an 8-bit number.

Algorithm:

- 1) Assigns source index a value to point at an address '0100:1000h' in RAM.
- 2) Moving CX register to source index and fetching the 8-bit number in the register.
- 3) Initialising AX register with value 0001h, to set a base for our answer.
- 4) Now loop "mult" will run until CX reaches value 1.
- 5) In the loop, CX will get multiplied by AX and value will be stored in AX register.
- 6) For After CX reaches 1, program will continue, moving forward with source index.
- 7) Now value obtained in AX register will be moved to the memory.
- 8) AH will be stored first and AL next to it, just to give it an order. ([si+15] is only to print them in next line, just for more clarity).
- 9) Hence, we obtain our desired result, and program ends with halt command.

Code:

; add your code here

```
mov si, 1000h
mov CX,[si]
mov AX,0001
mult:
mul CX loop mult
inc si
mov [si+15],AH
mov [si+16],AL
hlt
```

Output:

The screenshot displays a debugger interface with three main windows:

- Random Access Memory:** Shows a memory dump starting at address 0100:1000. The data is mostly zeros, with some non-zero values at higher addresses (e.g., 0100:1010 contains 02 D0).
- emulator: noname.bin:** The main debugger window. It includes a menu bar (file, math, debug, view, external, virtual devices, virtual drive, help), a toolbar with buttons like Load, reload, step back, single step, and run, and a step delay slider. The registers window on the left shows the state of various registers, with SI and DI highlighted. The instruction window on the right shows the current instruction being executed: `hlt` at address 0100:0013.
- original source co...:** A window showing the original assembly code, with the `hlt` instruction highlighted in yellow.

(ARKAJYOTI 2K19/EP/022)