University of Information Technology & Sciences

Department of
Computer Science and Engineering

Lab Report-02

Course Title: Microprocessors and Microcontrollers Lab
Course Code: CSE-360

Submitted To

Md. Ismail
Lecturer
Of
CSE Department

Submitted By

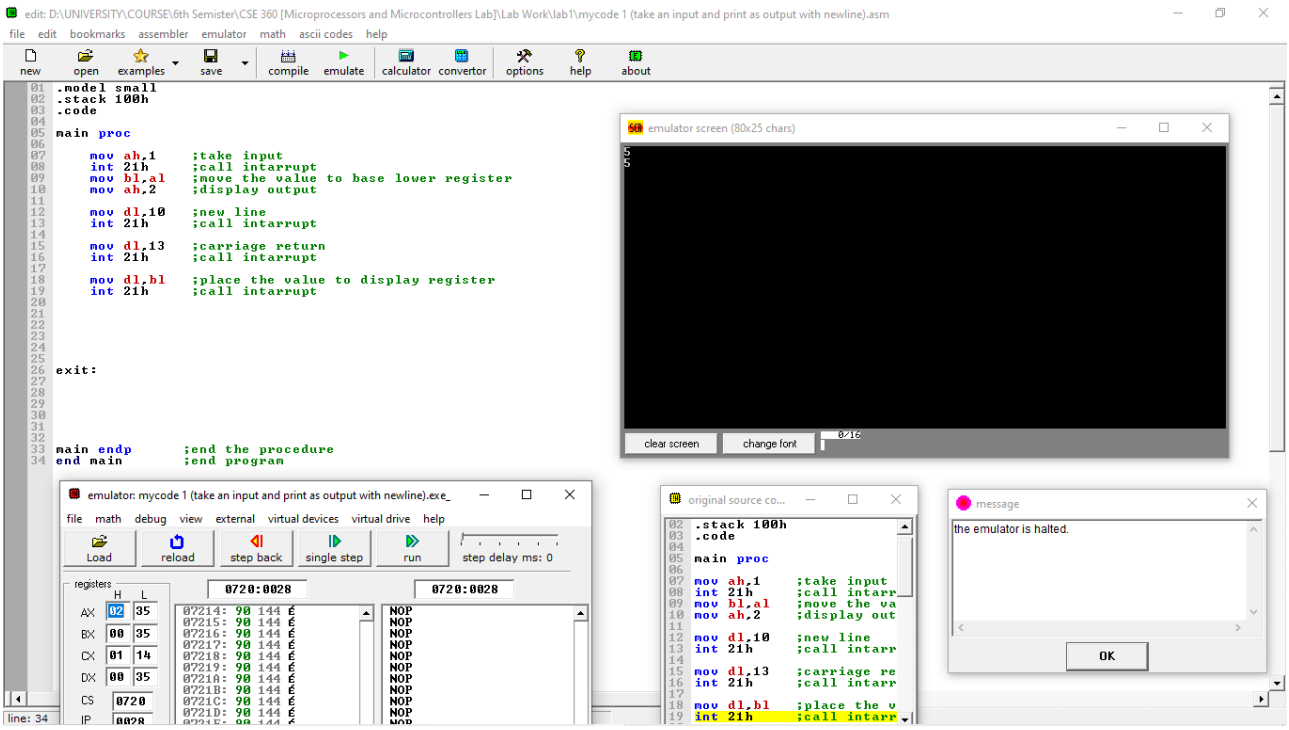
Name : Mobarok Hossain Zobaer
Id : 0432220005101029
Batch : 52
Semester: Spring 25
Section : 6A1

Problem Description: Write an assembly code to take an input and print it as output with newline

Implementation:

```
.model small
.stack 100h
.code

main proc

    mov ah,1    ;take input
    int 21h     ;call intarrupt
    mov bl,al   ;move the value to base lower register
    mov ah,2    ;display output

    mov dl,10   ;new line
    int 21h     ;call intarrupt

    mov dl,13   ;carriage return
    int 21h     ;call intarrupt

    mov dl,bl   ;place the value to display register
    int 21h     ;call intarrupt


exit:

main endp       ;end the procedure
end main        ;end program
```

Result:



Conclusion:

The assembly code provided demonstrates the process of taking a single character input from the user and displaying it back on the screen. The program uses DOS interrupt 21h services to achieve input and output operations. Here's a summary of the code's functionality:

Input (INT 21h, AH=1): The program first reads a character input from the user and stores it in the AL register.

Move Data (MOV BL, AL): The value from AL (input character) is then transferred to the BL register for later use.

New Line and Carriage Return (INT 21h): After the input, a newline (DL=10) and a carriage return (DL=13) are generated to format the output.

Output (INT 21h, AH=2): The program finally displays the character stored in BL by moving it into the DL register and calling the interrupt to output it to the screen.

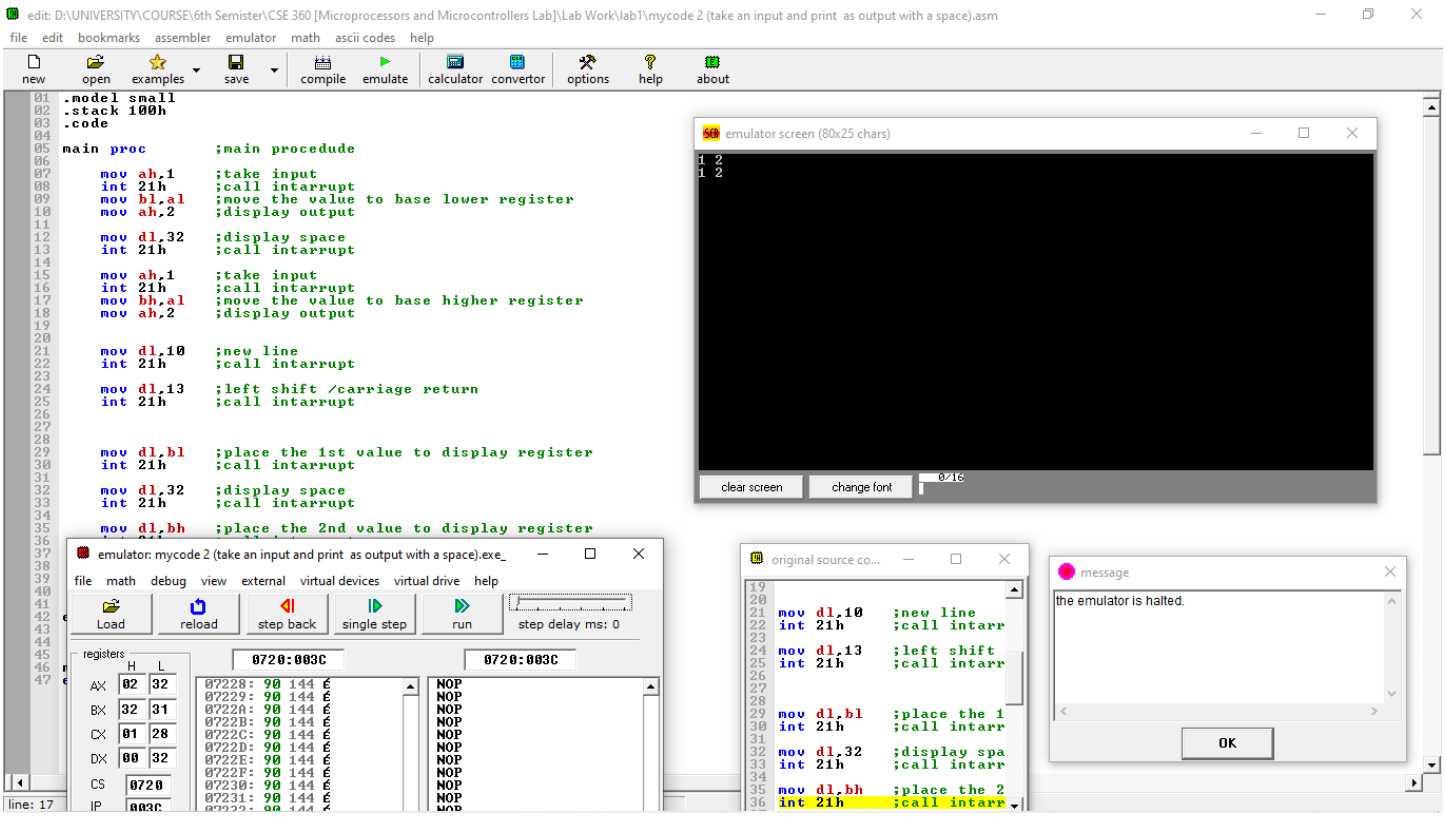
The program showcases simple input/output handling and demonstrates basic assembly operations like using interrupts, moving data between registers, and formatting output.

Problem Description: Write an assembly code to take an input and print it as output with a space

Implementation:

```
.model small
.stack 100h
.code

main proc       ;main procedude

    mov ah,1    ;take input
    int 21h     ;call intarrupt
    mov bl,al   ;move the value to base lower register
    mov ah,2    ;display output

    mov dl,32   ;display space
    int 21h     ;call intarrupt

    mov ah,1    ;take input
    int 21h     ;call intarrupt
    mov bh,al   ;move the value to base higher register
    mov ah,2    ;display output


    mov dl,10   ;new line
    int 21h     ;call intarrupt

    mov dl,13   ;left shift /carriage return
    int 21h     ;call intarrupt



    mov dl,bl   ;place the 1st value to display register
    int 21h     ;call intarrupt

    mov dl,32   ;display space
    int 21h     ;call intarrupt

    mov dl,bh   ;place the 2nd value to display register
    int 21h     ;call intarrupt

exit:

main endp      ;end the procedure
end main       ;end program
```

Result:



Conclusion:

This assembly program demonstrates how to take two characters as input from the user, store them in different registers, and then display them back in a formatted manner. Here's a brief breakdown of the operations performed:

Input (INT 21h, AH=1): The program first accepts a character from the user and stores it in the AL register, then moves it to the BL register.

Space (INT 21h, DL=32): A space is displayed between the two inputs for better formatting.

Second Input (INT 21h, AH=1): The program then accepts another character and stores it in the AL register, moving it to the BH register.

Output Formatting: The program displays a newline (DL=10) and a carriage return (DL=13) to organize the output properly.

Display (INT 21h, AH=2): It then displays both characters back to the screen—first the one stored in BL, followed by a space, and then the one stored in BH.

This program demonstrates basic input/output operations in assembly language, the use of registers for storing data, and formatting output with spaces and new lines to ensure the results are presented clearly.