|  |  |
| --- | --- |
| .Description: C:\Documents and Settings\MMAI\Desktop\EWULogo.png | **Department of Electrical and Electronic Engineering**  **EEE 401**  **MICROPROCESSORS & INTERFACING** |
|  |  |

**EXPERIMENT NO: 02**

## *Experimental Study of Input/ Output functions in Assembly Language*

**1.1 OBJECTIVE**

In this experiment, the assembly language of Intel 8086 microprocessor will be observed where students will learn to achieve input & output in emu 8086 and get the outputs.

**1.2 Pre- lab Preparation**

* Read the experiment thoroughly and make a real effort to answer the questions of pre-lab.
* Review 8086 System commands from Microprocessor internal lab before you coming to the lab.
  1. **Equipments**
* Personal Computer
* Emu 8086 Software

**1.4 Theoretical background:**

***System Commands***

Table 1 lists the most of the system commands used in the MTS-8088. The 8086 System has two command groups. These are the system commands and the I/O Driver commands. The system commands provide ways to utilize the system’s resources, and the I/O Driver Commands are used to control I/O devices.

|  |  |
| --- | --- |
| Command |  |
| **Memory Management Commands** | |
| D | Display the contents of Memory |
| C | Compare the contents of Memory |
| E | Edit/Modify the memory contents |
| F | Fill memory |
| M | Move the contents of memory |
| **Assembler Commands** | |
| A | Command A is used to write an assembly language program. |
| I | Insert instruction in the program |
| U | Disassemble the assembly language instructions into machine code |
| **Program Control Commands** | |
| G | Executing Programs |
| R | Display / modify the contents of registers. |
| T | Trace the program execution |
| **Numerical Operation/Conversion** | |
| B | Convert a decimal number into binary |
| H | Sum or difference of two hexadecimal numbers |
| J | Convert a decimal number in Hexadecimal |
| S | Convert a hexadecimal number into decimal |
| V | Convert a binary number into decimal |

* 1. **Procedure**
* **Part-1 : Input from keyboard and output on screen**
* Go to Start menu and run emu 8086 software.
* At first, you need to declare the library function.
* Then, you need to declare the main function like the following figure 1. Then, write down the code.

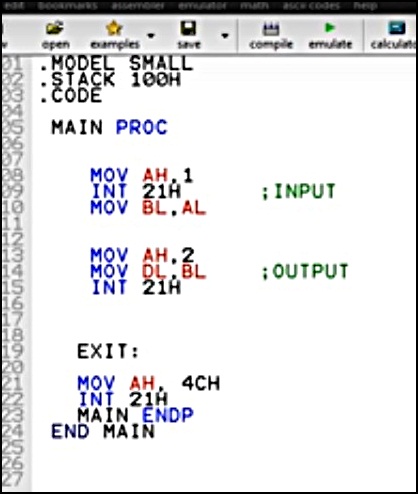


Figure: 01

* As you can see**,** the 1st part of the program the code consist of input properties and 2nd part consist of output properties.
* After you emulate the code, you need to press the RUN button, where you will see an input and a copy of it as an output. But consecutively. To omit this, you need to add the following properties of NEW\_LINE, as shown in the figure 02.

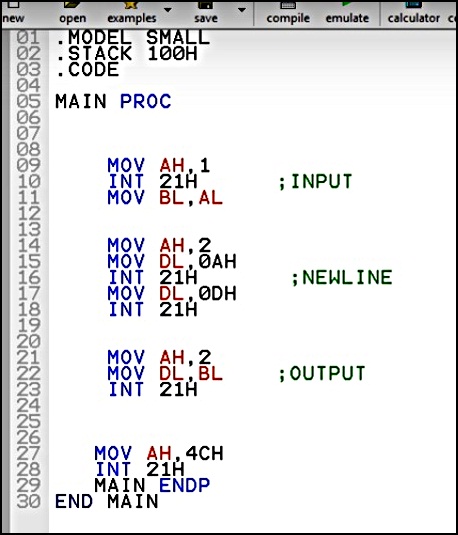
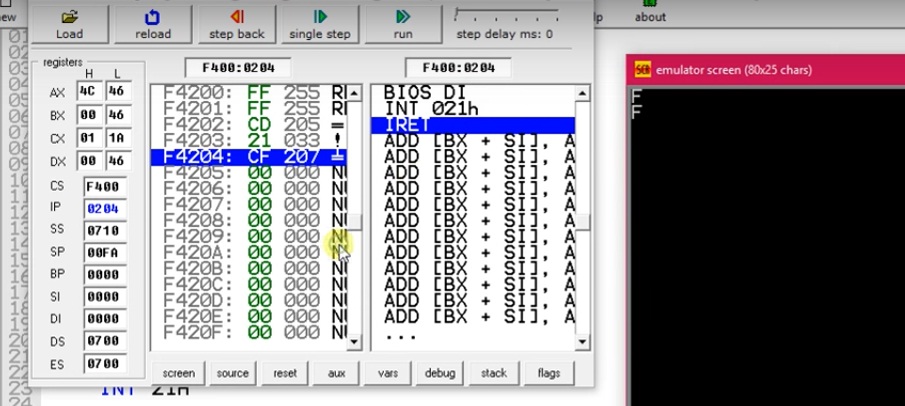
****

Figure: 02

* After you emulate the code, you need to press the RUN button, where you will see an input and a copy of it as an output but in the next line like figure 03.

 Figure: 03

* **PART -2: String Input output**
* Please follow what we have done in part 1. Here, you need to get a string output. As you can see in figure 04, we have used DX instead of DL, since we are using a string or DB i.e. double byte. The 2nd paragraph consists of print function which starts from load effective address command. Also, in part -1 we have used AH, 2 since it was only a integer value. But in this case, we have used MOV AH, 9; since it is a string.

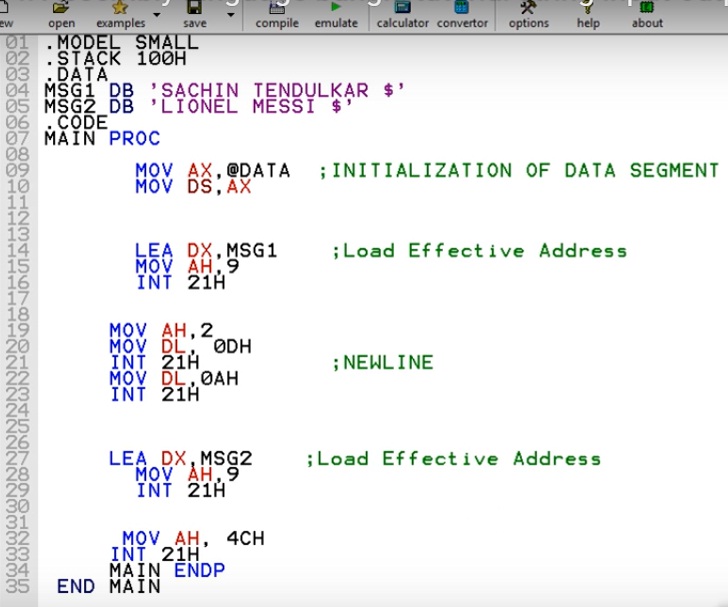
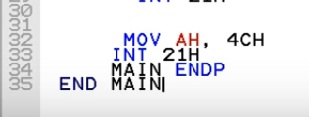
****

Figure: 04

* After you emulate the code, you need to press the RUN button. You will see an output like figure 05.

 Figure: 05

* You need to remember that every time we are using the following commands



as an end function. You can get output, without writing these lines. But it is a good practice, so that is why we have added in 2nd part as well.

* **Part-1 : Input from keyboard and output on screen**
* Go to Start menu and run emu 8086 software.
* At first, you need to declare the library function.
* Then, you need to declare the main function like the following figure 1. Then, write down the code.

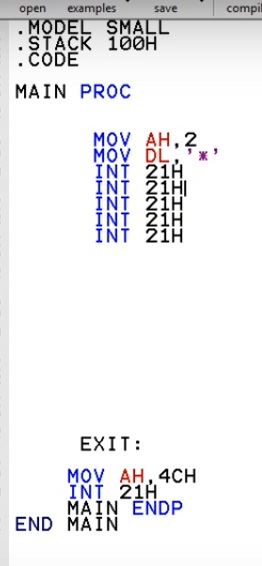


Figure: 01

* After you emulate the code, you need to press the RUN button, where you will see \*only number of times. Save the screenshot in your computer.
* But if you want to print the star \* number of time, we can use INT 21h command number of times. But instead of using INT 21h consecutively you need to add the following properties of JMP: jump command, as shown in the figure 02.

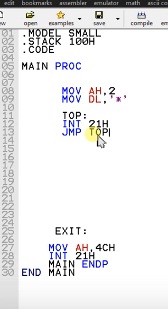
****

Figure: 02

* After you emulate the code, you need to press the RUN button, where you will see the star is printing repeatedly.
* To stop this, add a command line MOV CX,5 before the command line MOV AH,2 for acquiring printing control. You will see star is printed 5 times, since CX=5 (Count register)
* But, you have to declare the LOOP command like figure 03.
* If the user wants to take control of the star output, you have to make provision for taking user input. So, that we need to declare a function before the loop command like figure 04.

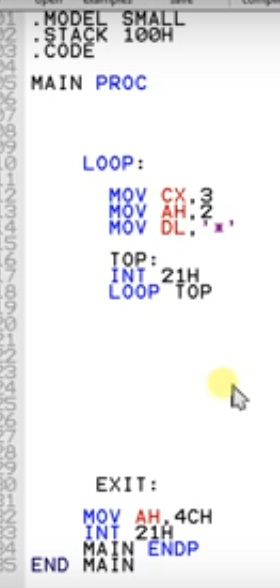


Figure 03

* **PART -2: Input output**

* For getting stars equal to ASCII values, e.g. pressing 1 and getting 49 stars, you need to add three command lines:

MOV AH, 1

INT 21H

MOV BL, AL

* After you emulate the code, you need to press the RUN button. You will see an output like figure 04.

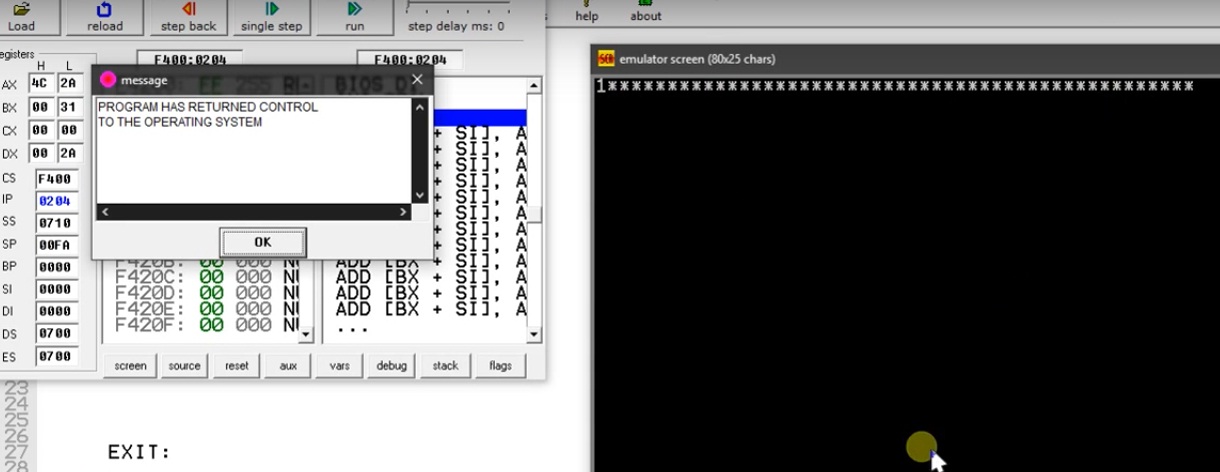
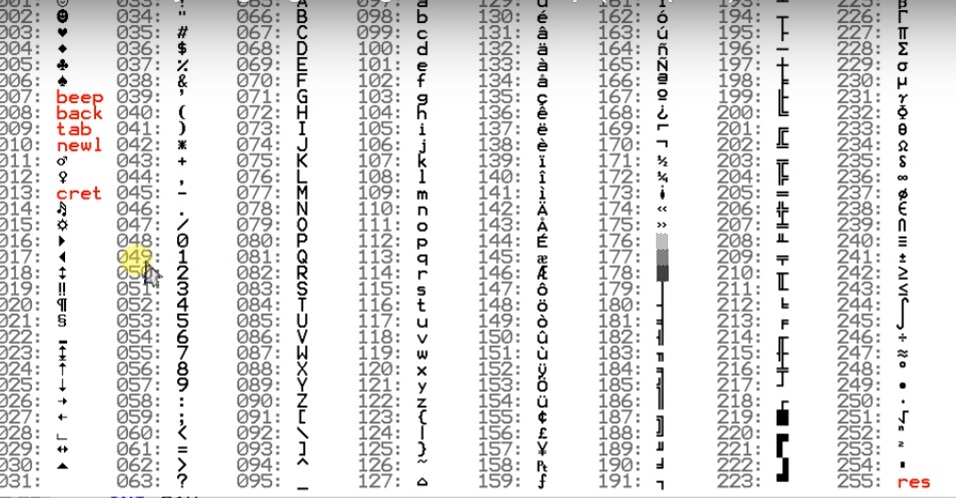


Figure: 04

* As you can see, instead of printing only one \*, we are getting \* for 49 times, but at least it is getting stopped, which was not happening previously.
* The reason it is happening because, if you see the ASCII table in figure 06, ASCII value of 1 is 049. So, as a consequence, if we press 2, we will get 50 stars.

 Figure: 05

* So, if we need to print only one star, after pressing 1, we have to deduct 49 from 48, like figure 06.

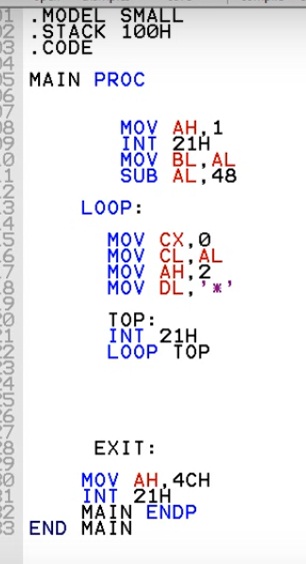


Figure: 06

* After the executing the above code, you will get the number of stars equal to the number you have pressed. This is how a user takes control of an output in a print function.
  1. **Post Lab Work:**
* Take the screenshots, after executing each command.
* Randomly chose any two strings for getting a different set of outputs for part-2.
* Discuss all the steps in your own words, preferably with screenshots of each step.