Lab # 7

# OBJECTive

Explain I/O operations of Intel 8086 Architecture

# Theory

**INT 21h:**

This can be used to invoke large number of DOS functions. A particular function is requested by placing a function number in the AH register and invoking INT 21h.

For E.g. following functions can be used in our program:

**Function No.Routine**

1h Single key input

**Examples:**

**Single Key Input**

MOV AH,01H **; input key function**

INT 21h

**MACROS**

A macro is a block of code that has been given a name and when the compiler encounters such name, it expands the macro, copies the block of code at the place of invocation into the program. The text may contain instructions or references to other macros.

The basic advantage of using a macro is to create new instructions. These macros can be a part of the program or they can be written in a separate file and that file can be included in the program. There is a list of useful macros which are helpful to be kept in the macro library of the assembly language compiler and these can be used in other user programs.

These can also be written in the lines of code of the program without any name given to them.

**Example:**

**CARRIAGE RETURN and LINE FEED**

This macro can be used to feed a new line in the output and is very useful as a new line is needed many times during a formatted output.

**NEW\_LINE MACRO**

**MOV AH, 02H**

**MOV DL, 0DH**

**INT 21H**

**MOV DL, 0AH**

**INT 21H**

**ENDM**

Once declared and defined, this macro can be invoked anywhere in the program to insert a new line. The name of the macro can be any string as defined by the user.

**Function No.Routine**

09h Character String Output

**Example**:

**Character String Output**

**MOV AH, 09H ; input key function**

**MOV DX, OFFSET MSG ; moves the offset of the msg to the register**

**INT 21h**

**STRINGS:**

A string is a list of characters treated as a unit. In programming languages we denote a string constant by using quotation marks, e.g. “Enter first number”.

In 8086 assembly language, single or double quotes may be used.

**Defining String Variables**

The following 3 definitions are equivalent ways of defining a string ‘abc’:

version1db “abc” ; string constant

version2 db ‘a’, ‘b’, ‘c’; character constants

version3 db 97, 98, 99; ASCII codes

The first version uses the method of high level languages and simply encloses the string in quotes. This is the preferred method.

The second version defines a string by specifying a list of the character constants that make up the string.

We may also combine the above methods to define a string as in the following example: message db “Hello world”, 13, 10, ‘$’

**STRING OUTPUT**

MS-DOS provides subprogram number **9h** to display strings which are terminated by the **‘$’** character. In order to use it we must:

1. Ensure the string is terminated with the ‘$’ character.
2. Specify the string to be displayed by storing its address in the **dx** register.
3. Specify the string output subprogram by storing 9h in ah.
4. Use int 21h to call MS-DOS to execute subprogram 9h.

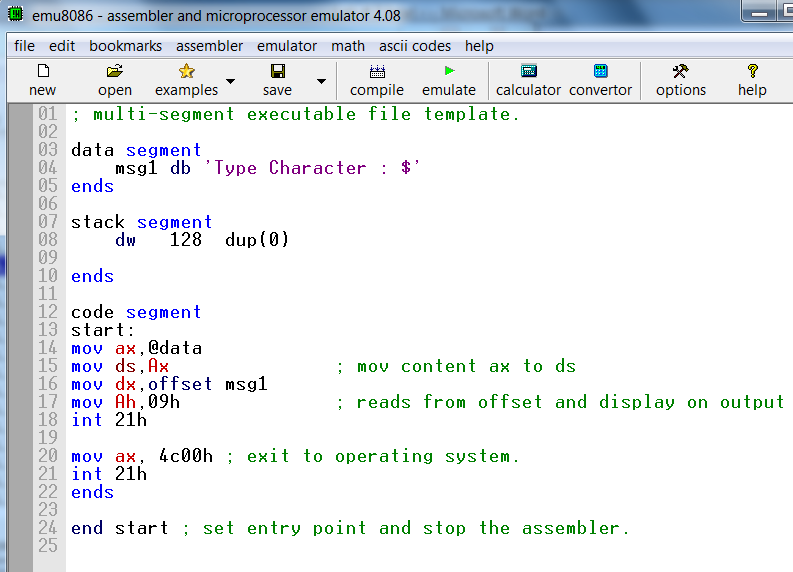
**Keyword: MSG**

The message to be displayed in the output can also be declared in the data segment using the keyword MSG, moving the string in the register DB and can be used afterwards.

MSG DB ‘HELLO!$’

The “$” marks the end of the string and is not displayed. If the string contains the ASCII code of a control character, the control function is performed.

**Sample Program**



**EXERCISE:**

Task#1:

Write a program that takes a single character input and displays it in a new line.

Task#2:

Write a program to display your bio data using carriage return and line feed macro

Task#3:

Write a program to make your 3rd semester mark sheet