



DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING

Title: Implement Macro in Assembly Language Programming

MICROPROCESSORS AND MICROCONTROLLERS LAB
CSE 304



GREEN UNIVERSITY OF BANGLADESH

1 Objective(s)

- To understand 8086 instructions related to Macro using Assembly Language Program.

2 Problem analysis

In the context of programming, a macro is a set of instructions that has been named and can be called with that name. It's a way to **reduce redundancy in your code** and make it more readable and maintainable. In assembly language programming, a macro is a sequence of instructions that has been given a name and can be called with that name. When the assembler encounters the macro call, it replaces the call with the entire sequence of instructions defined by the macro. Macros are typically used to define reusable code snippets or to create shorthand for commonly used sequences of instructions. They help in writing more modular and readable code, as you can use a simple macro call instead of writing out the same set of instructions multiple times.

Syntax for Declaring MACRO:

```
macroName MACRO d1, d2, . . . , dn
Statement 1
Statement 2
. . .
Statement k
ENDM
```

Where d1, d2, . . . ,dn is an optional list of dummy parameters. Dummy parameters are temporary variables; they are not declared by data definition directives (DB, DW). They can be used as input as well as output parameters.

A macro definition can appear anywhere in an assembly language program before the END directive. It is usual to place all macro definitions at the beginning of a program before the segment definitions.

The difference between macro and procedure is that procedure requires separate memory location to store, where as macro uses the same memory location of the code where the macro actually is being called. Furthermore, in macro parameter values can be passed.

3 Assembly Language Program to Add Two Numbers using Macro

```
1  ADDITION  MACRO NUM1, NUM2           ; Declaration of Macro
2      MOV AX, NUM1
3      MOV BX, NUM2
4
5      ADD AX, BX
6  ENDM
7
8  ORG 100H
9
10 .DATA
11     A DW 10
12     B DW 5
13
14 .CODE
15     MAIN PROC
16
17         ADDITION A, B                 ; Calling Macro
18
19
20     MOV AH, 4CH
21     INT 21H
```

```

22
23     MAIN ENDP
24 END MAIN

```

4 Sample Input/Output (Compilation, Debugging & Testing)

The program will store the addition of two numbers in AX register.

5 Assembly Language Program to Print String using Macro

```

1 MDSPLY_STRING MACRO STRING      ; Declaration of MACRO
2     MOV DX, OFFSET STRING
3
4     MOV AH, 09H
5     INT 21H
6 ENDM
7
8 ORG 100H
9
10 .DATA
11     MESSAGE1 DB 'Microprocessors and $'
12     MESSAGE2 DB 'Microcontrollers Lab$'
13
14 .CODE
15     MAIN PROC
16         MOV AX, @DATA
17         MOV DS, AX
18
19         MDSPLY_STRING MESSAGE1 ; 1st Call of the MACRO
20         MDSPLY_STRING MESSAGE2 ; 2nd Call of the MACRO
21
22         MOV AH, 4CH
23         INT 21H
24
25     MAIN ENDP
26 END MAIN

```

6 Sample Input/Output (Compilation, Debugging & Testing)

The program will print two string using macro named MDSPLY_STRING.

Microprocessors and Microcontrollers Lab

7 Discussion & Conclusion

Based on the focused objective(s) to understand about macro in assembly language programming, the additional lab exercise made me more confident towards the fulfilment of the objectives(s).

8 Lab Task (Please implement yourself and show the output to the instructor)

1. Write an Assembly Language code that takes an input ARRAY and passes the array values and address to a MACRO. Using the array, address and one procedure separate out the ODD digits and EVEN digits.

Input:
2 0 4 7 1 9
Output:
ODD Digits: 7 1 9
EVEN Digits: 2 0 4

9 Lab Exercise (Submit as a report)

- Write an Assembly Language code that takes an input ARRAY and passes the array values and address to a MACRO. Now produce the summation of odd digits and even digits as output.

Input:
3 1 4 5 1 6 8 7
Output:
ODD Digits: 17
EVEN Digits: 18

10 Policy

Copying from internet, classmate, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.