



Lecture – 4



Section-B

Macro Language and Macro processor

References: John. J. Donovan




Introduction

- Macro Instructions

Introduction

- ▶ The assembly language programmer often finds it necessary to repeat some blocks of code many times in the course of a program.
- ▶ The block may consist of code to save or exchange set of registers, for eg. Or code to set up linkages or perform a series of arithmetic operations.
- ▶ In this situation, the programmer will find a macro instruction facility useful.
- ▶ **Macro instructions** (often called **macros**) are single-line abbreviations for groups of instructions. In employing a macro, the programmer essentially defines a single “instruction” to represent a block of code.

- 
- Macro instructions are usually considered an extension of the basic assembler language, and the macro processor is viewed as an extension of the basic assembler program.
 - As a form of programming language, however, macro instruction languages differ significantly from assembly language and compiled algebraic

Macro Instructions

▶ In its simplest form, a macro is an abbreviation for a sequence of operations. Consider the following program:

▶ Example 1:

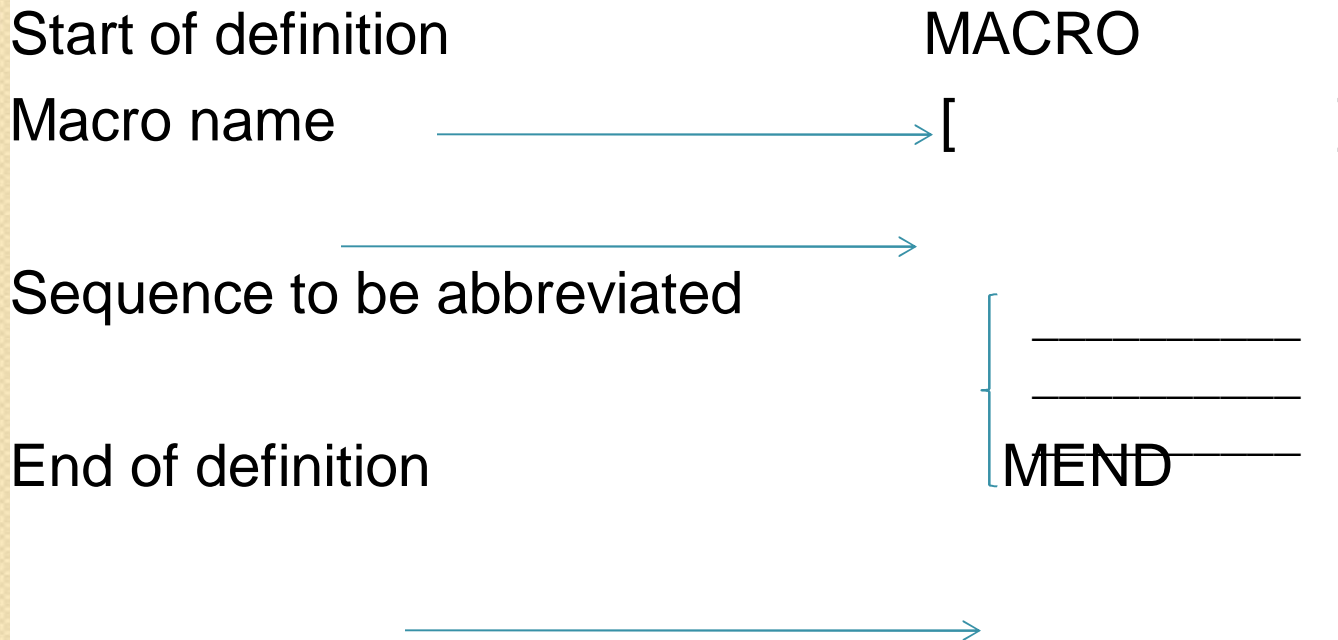
```
.  
. .  
A      1, DATA      Add contents of DATA to register 1  
A      2, DATA      Add contents of DATA to register 2  
A      3, DATA      Add contents of DATA to register 3  
. .  
A      1, DATA      Add contents of DATA to register 1  
A      2, DATA      Add contents of DATA to register 2  
A      3, DATA      Add contents of DATA to register 3  
. .  
DATA   DC           F'5'
```


In the above program the sequence

```
A      1, DATA  
A      2, DATA  
A      3, DATA
```

occurs twice

- A macro facility permits us to attach a name to this sequence and to use this name in its place
- A macro processor effectively constitutes a separate language processor with its own language.
- We attach a name to a sequence by means of a macro instruction definition, which is formed in the following manner :



- 
- The MACRO pseudo-op is the first line of the definition and identifies the following line as the *macro instruction name*.
 - Following the name line is the sequence of instructions being abbreviated – the instructions comprising the “macro” instruction.
 - The definition is terminated by a line with the MEND (“macro end”) pseudo-op.
 - Once the macro has been defined, the use of the macro name as an operation mnemonic in an assembly program is equivalent to the use of the corresponding instruction sequence.
 - Our example might be rewritten as follows, assigning the name “MOR” to the repeated sequence



Source

MACRO

INCR

A

A

A

MEND

.

.

.

INCR

.

.

.

INCR

.

.

.

DATA

DC

.

.

.

1,DATA

2, DATA

3, DATA

F'5'

Expanded Source

.

.

.

A

1,DATA

A

2,DATA

A

3. DATA

.

.

.

A

1,DATA

A

2,DATA

A

3. DATA

DATA

DC

F'5'

- ▶ In this case the macro processor replaces each macro call with the lines

A	1, DATA
B	2, DATA
C	3, DATA

- ▶ This process of replacement is called *expanding* the macro.
- ▶ Notice that the macro definition itself does not appear in the expanded source code.
- ▶ The definition is saved by the macro processor. The occurrence in the source program of the macro name, as an operation mnemonic to be expanded, is called a macro call.



Features of Macro Instructions

- To be continued....



Applications and Scope of Research

Will be discussed in next lecture, after the completion of the topic : Macro Instructions !!