Roll No: 33231
Name: Prajakta Keer

ASSIGNMENT 3

Title: Macro Processor

Problem Statement: Study assignment for macro processor (consider all aspects of macro processor)

Objectives:
- To study basic functioning of macro pre processor
- To study working of nested macros

Theory: What is macro processor?

- Macro represents a group of commonly used statements in the source programming language.
- Macro processor replaces each macro instruction definition with the corresponding group of source language statements. This is known as expansion of macros.
- Using macro instructions programmers can leave the mechanical details to be handled by the macro preprocessor
- Macro processor designs are not directly related to the computer architecture on which it runs.
- Macro processor involves definition, invocation and expansion

MACRO DEFINITION

- MACRO - identifies the beginning of macro definition
- MEND - identifies the end of macro definition

Eg:
```
MACRO
INCRA & MEM_VAL & INC_VAL, &REG
MOVER &REG, &MEM_VAL
ADD &REG &INC_VAL
MOVEM &REG &MEM_VAL
MEND
```

Macro Call :- INCR A, B, AREG

Parameter Passing Methods in macro

(A) Positional argument or parameter argument are matched with dummy arguments according to their order in which they appear or compute the ordinal position of the formal parameters.

```
INCR A, B, AREG
```

A replaces 'first' dummy arg.
B replaces 'second' dummy arg. and so on.

(B) Keyword Parameters : They are useful where macros use a long list of parameters

Format : & <name_of_par> = <actual_par>

Eg : MNAME par1 = REG1, par2 = REG2

(C) Keyword Defaut Parameters : If a parameter has the same value in most calls on a macro, this value can be specified as its default value in the macro definition ~~table~~ itself

Format : & <param name> [<param kind>] [<default value>]

Eg :
```
MACRO
    MNAME &par1 = REG1, &par2 = , &par3 =
    MOVER &par1, &par2
    ADD &par1, &par3
    MOVEM &par1, &par2
    MEND
```

MACRO CALL : MNAME par2 = A, par3 = B OR
MNAME par1 = REG2, par2 = A, par3 = B

(D) Mixed Parameter : A macro definition may use both positional and keyword parameter. In such a case, all positional parameters must precede all keyword parameters in a macro call

Eg :
```
MACRO
    MCALC &par1 = , &par2 = , &par3 = MULT, &par4 = &par4
    ADD REG1, &par2
    MEND
```

Macro Call : MCALC A, B, par4 = loop

33231

* Data structures needed for macroprocessor

1) MDT (Macro Definition Table) : • stores macro definition including macro prototype and macro body
• References to the macro instruction parameters are converted to a positional notation for efficiency in substituting arguments.

2) MNT (Macro Name Table) : stores macro names which serves as an index to the MDT • contains pointers to the beginning and end of the definition.

3) ALA (Argument List Array) : • Used during the expansion of macro invocations • When a macro invocation statement is encountered, the args are stored in this table according to their position in the argument list.

* Nested Macro (macro) Call : A macro may constitute a call on another ~~means~~ • This is known as nested macro call.
It has the following types :

1) Call within macro definition : when a macro is called/ invoked inside the definition of another macro.

eg:
```
MACRO                MACRO
ADD &A               ADDS &A1, &A2, &A3
L 1,&A               ADD &A1      } Macro ADD invoked inside
MEND                 ADD &A2      } definition of Macro
                     ADD &A3        ADDS
                     MEND
```

2) Definition within definition : Macro definition within another macro definition • The inner macro definition is processed during the outer macro expansion.

Outer macro definition
[Inner macro definition]

Eg:
```
MACRO
ABC &MAIN
MACRO
   &MAIN &W
   ADD AREG ,&W
MEND
MOVEM BREG, ='S'
MEND
  :
ABC HELLO
HELLO X
X DC 'S'
END
```

Inner macro → Inner macro definition is not defined until the outer macro has been called.

Outer macro →

* Advanced Macro facilities:

- Facility support at the time of semantic expansion
- Facilities for alteration of flow of control during expansion. Flow is altered within 2 features:
  1) Expansion time sequencing symbol (SS)
  2) Expansion time statement AIF, AGO, ANOP

AIF <expression> <sequencing symbol>
AGO <sequencing symbol>
<sequencing symbol> ANOP

Conclusion: I studied basic functionality of macro processors and also working of nested macros.