

Assignment 3

Macro Processor

→ Problem Statement :

Study assignment for macro processor. Consider all aspects of macro processor.

→ Objectives :

1. To study basic functioning of Macro Processor
2. To study working of nested macro

→ Theory

A] What is a macro processor?

A macro instruction is the notational convenience for the programmer. For every occurrence of macro the whole macro body or macro block statements get expanded in the main source code. Thus macro instructions make writing code more convenient.

A macro definition consists of following parts:

1. Macro name
2. Start of definition
3. Sequence of statements
4. End of definition.

It follow a structure like :

```
MACRO  
MACRO_NAME  
:  
:  
:  
:  
:  
:  
MEND
```

} Body of Macro

Example: MACRO
MAC1
MOVER AREG, M
ADD BREG, M
MOVEM CREG, M
MEND

B] Parameter Passing Methods :

There are 2 types of macro parameters :

1] Positional Parameters :

They are used to assign values ~~used~~ based on their position in the macro definition and at invocation. The order that you use to specify the values must match the order they are listed in macro definition.

2. Keyword Parameters :

The benefit of using keyword parameters is the ability to give macro variables a default value within the macro definition. When you assign values using keyword parameters, you must include an equal sign after the macro variable name.

● [C] Pass Structure of Macro Processor :

1. Two-Pass Macro Processor :

Like an assembler or a loader, we can design a 2-pass macro processor in which

- First Pass processes all macro definitions
- Second Pass expands all macro invocation statements.

2. One-Pass Macro Processor

- A One-Pass Macro Processor that alternate between macro definition & macro-expansion is able to handle "macro-in-macro"
- However, because of one-pass structure, the definition of a macro must appear in the source program before any ~~system~~ statements that invoke that macro.

D] Data Structures required :

1. The macro names are entered into the NAMTAB, NAMTAB contains two pointers to the beginning & end of the definition in DEFTAB.
2. The DEFTAB stores all the macro definitions.
3. The third data structure is the argument table ARGTAB, which is used during the expansion of invocations.
4. The arguments are stored in ARGTAB according to their position in the argument list.

E] Advanced Macro Facilities

1. Nested Macros :

- When a macro definition is being entered into DEFTAB, the normal approach is to continue until MEND directive is reached.
- This will not work for "macro in macro" because the MEND ~~with~~ 1st encountered (for inner macro) will terminate the whole macro definition process.
- To solve this, a counter LEVEL is used to keep track of the ~~the~~ level of macro definitions.

2. Facilities for alternating flow of control during expansion :

Statements like AIF, AGO & ANOT can be used to conditionally & unconditionally transfer the ~~ff~~ flow of control to the target memory address -

Syntax :

- AIF → AIF (<expression>) <Label sequential symbol>
- AGO → AGO <sequential symbol>
- ANOT → <Sequential Symbol> ANOT

3. Expansion Time Variable :

They are used during expansion of macros. A local EV is created for use inside a particular MACRO. A global EV exists across all macro calls.

- LCL <EV specification>
- GBL <EV specification>

Ex. MACRO

CALL

· LCL $\&A, \&B$ \leftarrow creating 2 local variables
· $\&A$ SET 1
· $\&B$ SET 5
MEND

4. Attributes of formal parameters:

It represents information about the value of the formal parameters. These attributes are type, length and size that have names T, L & S.

Syntax:

<attribute name>'<formal parameter>

Ex. MACRO

CALL

· AIF (L' $\&A$ EQ 1) NEXT

NEXT: ..

·
·

MEND

Here expression control is transferred to NEXT only if length of A is equal to 1

→ Conclusion :

Topics Covered :

1. Macro Processor
2. Structure of Macros
3. Low Structures of Macros
4. Types of Parameters to call macros .
5. Advanced Macro facilities .