Paper [Jan-2011]

Chapter: 5. Macros & Macro processors

* Construct all data structures for Macro given below

```
MACRO
MCA     &X, &Y, &REG = BREG.
AIF     (&Y EQ 0)                .ERR.
MOVER   &REG, &X.
DIV     &REG, &Y.
.ERR    MEND.
```

Also generate the statements for these two Macro Call.

i. MCA 5, 5, REG = AREG.

ii. MCA 2, 0

Soln:

PNTAB

1	X
2	Y
3	REG

EVNTAB

1	-

SSNTAB

1	ERR

MNT

Name	#PP	#KP	#EV	MDTP	KPDTP	SSTP
MCA	2	1	0	25	5	5

KPDTAB

5	REG	BREG

SSTAB

5	25

MDT

25	AIF ((P,2) EQ 0) (S,1)
26	MOVER ((P,3)), (P,1)
27	DIV (P,3), (P,2)
	(S,1) MEND

i) MCA 5, 5, REG = AREG.

APTAB

5
5
AREG

EVTAB | 0 |

ii) MCA 2, 0

APTAB

2
0
BREG

EVTAB | 0 |

Data Structure of the Macro preprocessor.

[Jan-2011]

* Explain the Algorithm for Macro Expansion.

Ans:

Note: Read Answer from Book - Page No:- 153

[Jan-2011]

* Explain the procedure for expansion of Nested Macro Calls in detail. [07]

Ans:

1) A model statement in a macro may constitute a call on another macro is called nested macro calls.

2) Expansion of nested macro calls follows the last - in - first - out (LIFO) rule.

⇒ Example: Nested Macro Call in Macro.

```
MACRO
INCR_D    &MEM_VAL =, &INCR_VAL =, &REG = AREG.
MOVER     &REG , &MEM_VAL.
ADD       &REG, &INCR_VAL
MOVEM     &REG , &MEM_VAL.
MEND.
```

A Macro definition.

⇒ In above program INCR_D shows the expanded code for the call.

```
+    COMPUTE    X, Y.
```

⇒ After lexical expansion, the second model statement of COMPUTE is recognized to be a Call on macro INCR_D.

⇒ Expansion of this macro is now performed. This leads to generation of statement marked [2], [3] & [4] in following.

```
+    MOVEM    BREG, TMP
+    MOVER    BREG, X
+    ADD      BREG, Y
+    MOVEM    BREG, X
+    MOVER    BREG, TMP
```

```
MACRO
COMPUTE     &FIRST , &SECOND
MOVEM       BREG,  TMP
INCR_D      &FIRST, &SECOND, REG=BREG.
MOVER       BREG,  TMP
MEND.
```

A nested macro call.

```
                    + MOVEM  BREG, TMP [1]
                    +INCR_D  X, Y      + MOVER  BREG, X [2]
COMPUTE  X, Y.                         + ADD    BREG, Y [3]
                                       + MOVEM  BREG, X [4]
                    + MOVER  BREG, TMP [5]
```

Expanded Code for a nested macro call.

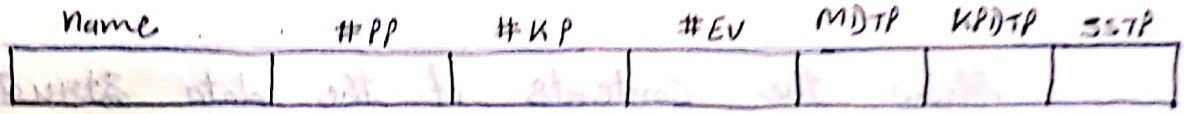
Dec - 2017

* Explain the entry format for the macro name table (MNNT)

Ans:

⇒ MNT has entries for all macros defined in a program. Each MNT entry contains three pointers MDTP, KPDTP & SSTP, which are pointers to MDT, KPDTAB & SSNTAB for the macro, respectively.

⇒ Formate of MNT table:



MNT.

Table.	Fields in each entry.
Macro name table (MNT.	- Macro name, - no. of positional parameter (#PP) - no. of keyword parameter (#KP) - No. of expansion time variable (# EV), - MDTP (Macro definition Table Pointer) - KPDTABP. (Keyword Parameter Default Table Pointer) - SSTABP (Sequential Symbol Table pointer).

Madhvi Bera

[June-2011]

* Give the following macro definition: [07]

```
        MACRO
        CLEARMEN    &X, &N, &REG = AREG.
        LCL         &M
&M      SET         0
        MOVER       &REG = '0'
.Mo     MOVEM       &REG, &X + &M.
&M      SET         &M + 1
        AIF         (&M NE &N) .Mo
        MEND
```

Show the contents of the data structures deployed by the macro-processor for the call.

```
        CLEARMEN    AREA, 10.
```

Ans: NOTE: Read answer from Book - Page No:- 151.

[June-2011]

* Explain data structures used for Macro processing. [07]

Ans: => To obtain a detailed design of the data structure it is necessary to apply the practical criteria of processing efficiency & memory requirements.

=> In Macro preprocessor, there are nine table generated during the execution process.

- PNTAB : Parameter Name Table.
- EVTAB : Expansion time Variable.
- SSNTAB : Sequential Symbol Name Table.
- MNT : Macro Name Table.
- KPDTAB : Keyword Parameter Default Table.
- SSTAB : Sequential Symbol Table.
- MDT : Macro Definition Table.
- APTAB : Actual Parameter Table.
- EVTAB : Expansion time Variable Table.

- PNTAB is used while processing the definition of a Macro., It contains formal parameter names.

- EV names are entered in EVNTAB while processing EV declarations.

- SS name are entered in SSTAB while processing an SS reference or definition.

- MNT has entries for all macros defined in a program. Each MNT entry contains three pointers MDTP, KPDTP, & SSTP which are pointer to MDT, KPDTAB & SSNTAB for the Macro, respectively.

- MDT entries are constructed while processing the model statements & preprocessor statement in the macro body.

Madhuri Rana

⇒ APTAB is constructed while processing a Macro call. & EVTAB is constructed at the start of expansion of a macro.

⇒ Example: Construct the data structure of Macro processor for following program.

```
        MACRO
        CLEARMEM   &X, &N, &REG = AREG.
        LCL        &M.
&M      SET        0
        MOVER      &REG , ='0'
.MORE   MOVEM      &REG , &X + &M.
&M      SET        &M + 1
        AIF        (&M NE N) .MORE.
        MEND
```

Macro Call CLEARMEM AREA, 10.

Soln:

PNTAB

1	X
2	N
3	REG

EVNTAB

M

SSNTAB

MORE

MNT

Name	#PP	#KP	#EV	MDTP	KPDTP	SSTP
CLEARMEM	2	1	1	25	10	5

Madhuri Bera

KPDTAB

10	REG	AREG

SSTAB

5	28

MDT

	LCL	(E,1)
(E,1)	SET	0
	MOVER	(P,3) = '0'
(S,1)	MOVEM	(P,3), (P,1) + (E,1)
(E,1)	SET	(E,1) + 1
	AIF	((E,1) NE (P,2)) (S,1)
	MEND	

APTAB

AREA
10
AREG

EVTAB

0

Data Structures of Macro preprocessor

[June-2011]

** In an assembly language program, a certain action is required at 10 places in the program. under what conditions would you code this action as. [04]

a. A macro ?

b. A Sub-routine ?

Ans:-

→ use of a macro name in the mnemonic field of

an assembly statement leads to its expansion, whereas use of a subroutine name in a call instruction leads to its execution.

⇒ Thus, programs using macros & subroutines differ significantly in terms of program size & execution efficiency.

⇒ Macros can be said to trade program size for execution efficiency of a program.

[Dec-2011]
* List and explain all the data structures used in Pass-I of Macro pre processor. [07]

Ans:-

NOTE:- Read answer from GTU Exam Paper ques. Solution. & Page No:- 6 (Material)

[Dec-2011]
* List and explain Pass-I algorithm of Macro pre processor. [07]

Ans:-

NOTE:- Read answer from Book - Page No:- 150

[May-2012]
* Define Macro.

A Macro is a unit of specification for program generation through expansion.

Madhvi Bera

May. 2012]

* Discuss the points to be taken care while designing a macro preprocessor. [07]

Ans:-

2) The following 4 step procedure is followed to arrive at a design specification for each task:

1. Identify the information necessary to perform a task.
2. Design a suitable data structure to record the information.
3. Determine the processing necessary to obtain the information.
4. Determine the processing necessary to perform the task.

2) Application of this procedure to each of the pre processor task is described in the following:-

* Identify macro calls :-

2) A Table called the macro name table (MNT) is designed to hold the names of all macros defined in a program.

2) A macro name is entered in this table when a macro definition is processed.

Madhwi Bera.

* Determine value of formal parameters :-

⇒ A table called the actual parameter table (APT) is designed to hold the values of formal parameters during the expansion of a macro call.

⇒ Each entry in the table is a pair.
(<formal parameter name>, <value>)

⇒ Two items of information are needed to construct this table, name of formal parameters, and default values of keyword parameters. This table is called parameter Default Table (PDT) is used for each macro.

* Maitain expansion time variables :-

⇒ An expansion time variables' table (EVT) is maintained for this purpose. The table contains pairs of the form.
(<EV name>, <value>)

⇒ The value field of a pair is accessed when a pre processor statement or a model statement under expansion refers to an EV.

* Organize expansion time control flow :-

⇒ The body of a macro, i.e. the set of pre processor.

Madhuri Bera

Statements & Model statements in it, is stored in a table called the macro definition table (MDT) for use during macro expansion.

2) The flow of control during macro expansion determines when a model statement is to be visited for expansion.

* Determine values of sequencing symbols :-

2) A sequencing symbols table (SST) is maintained to hold this information. The table contains pairs of the form.

(<sequencing symbol name>, <MDT entry #>)

Where <MDT entry #> is the no. of the MDT entry which contains the model statement defining the sequencing symbol.

* perform expansion of a model statement :-

i. MEC points to the MDT entry containing the model statement.

ii. Values of formal parameters & EV's are available in APT & EVT, respectively.

iii. The model statement defining a sequencing symbol can be identified from SST.

Madhvi Berra

11]

Explain Conditional Expansion with suitable example.

1) Conditional expansion helps in generating assembly code specifically suited to the parameters in a macro call.

2) This is achieved by ensuring that a model statement is visited only under specific conditions during the expansion of a macro.

3) The AIF & AGO statements are used for this purpose.

4) Example: For A = B + C.

```
        MACRO
        EVAL    &X, &Y, &Z.
        AIF     (&Y EQ, &X) .ONLY
        MOVER   AREG , &X
        SUB     AREG , &Y
        ADD     AREG, &Z
        AGO     .OVER
.ONLY   MOVER   AREG , &Z
.OVER   MEND
```

Madhuri Bera