INDIAN INSTITUTE OF TECHNOLOGY ROORKEE

Macro Processors Extensions

Macro Processors Design

- Two pass algorithm
  - Pass1: Recognize macro definitions
  - Pass2: Recognize macro calls and expand
- Nested macro definitions are not allowed

Nested Macros Definition

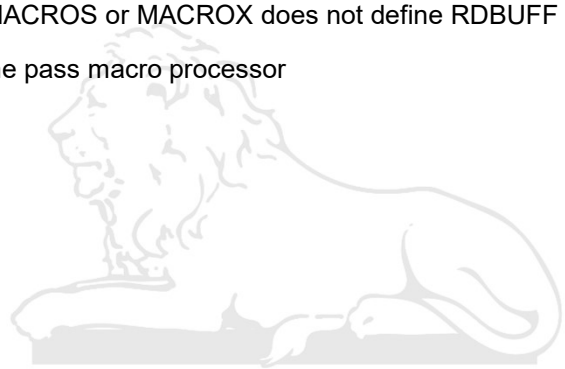
- Macro definition within macros
  - process macro definition during expansion time

```
MACROS   MACRO    {Defines SIC standard version macros}
RDBUFF   MACRO    &INDEV,&BUFADR,&RECLTH
         .
         .        {SIC  standard version}
         .
         MEND     {End of RDBUFF}
WRBUFF   MACRO    &OUTDEV,&BUFADR,&RECLTH
         .
         .
         .
         MEND
         .
         .
         .
         MEND
```

```
MACROX   MACRO    {Defines  SIC/XE macros}
RDBUFF   MACRO    &INDEV,&BUFADR,&RECLTH
         .
         .        {SIC/XE version}
         .
         MEND     {End of RDBUFF}
WRBUFF   MACRO    &OUTDEV,&BUFADR,&RECLTH
         .
         .
         .
         MEND
         .
         .
         .
         MEND
```

Two pass algorithm

- Defining MACROS or MACROX does not define RDBUFF / WRBUFF
- Design One pass macro processor



IIT ROORKEE

Generation of Unique Labels

- Example
  - JEQ *-3
  - inconvenient, error-prone, difficult to read
- Example
  - $LOOP TD =X'&INDEV'
    JEQ $LOOP
  - 1st call:
    - $AALOOP TD =X'F1'
      JEQ $AALOOP
  - 2nd call:
    - $ABLOOP TD =X'F1'
      JEQ $ABLOOP

RDBUFF F1, BUFFER, LENGTH

```
25   RDBUFF   MACRO   &INDEV,&BUFADR,&RECLTH
30            CLEAR   X
35            CLEAR   A
40            CLEAR   S
45           +LDT     #4096
50   $LOOP    TD      =X'&INDEV'
55            JEQ     $LOOP
60            RD      =X'&INDEV'
65            COMPR   A,S
70            JEQ     $EXIT
75            STCH    &BUFADR,X
80            TIXR    T
85            JLT     $LOOP
90   $EXIT    STX     &RECLTH
95            MEND
```

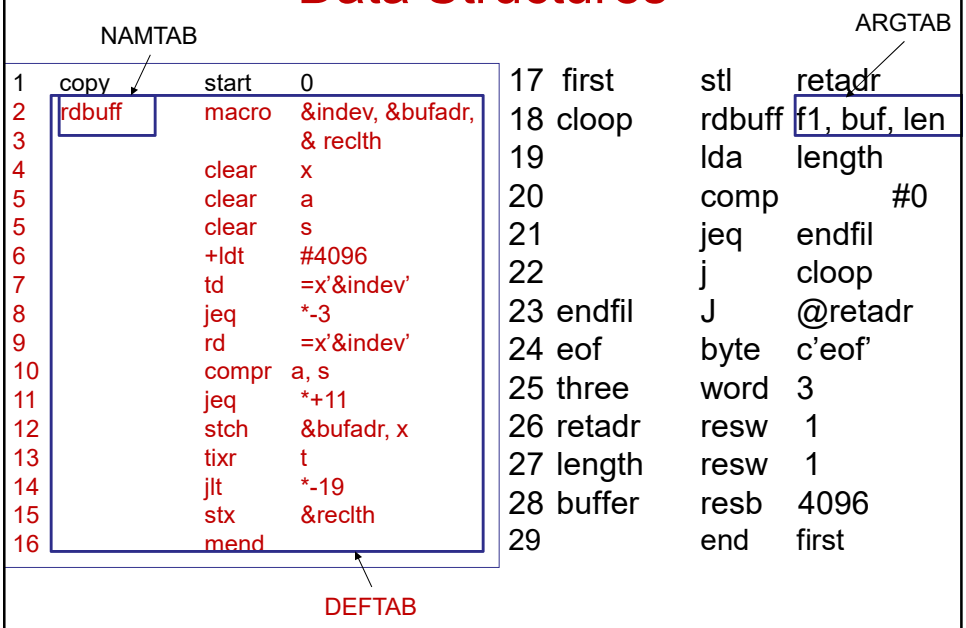
```
30            CLEAR   X
35            CLEAR   A
40            CLEAR   S
45           +LDT     #4096
50   $AALOOP  TD      =X'F1'
55            JEQ     $AALOOP
60            RD      =X'F1'
65            COMPR   A,S
70            JEQ     $AAEXIT
75            STCH    BUFFER,X
80            TIXR    T
85            JLT     $AALOOP
90   $AAEXIT  STX     LENGTH
```

One-Pass Macro Processor

- Definition of a macro must appear before its invocation
- Alternates between macro definitions and macro expansions
- Data structures: DEFTAB, NAMETAB, ARGTAB

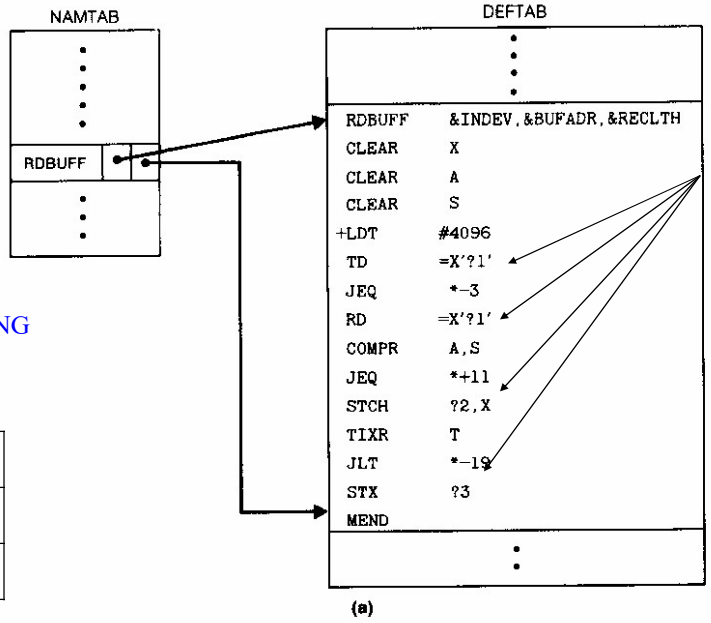
- Procedures
  - Main
    - GETLINE
    - PROCESSLINE
  - macro definition: DEFINE
  - macro invocation: EXPAND

Data Structures



Data Structures -- Global Variables

- DEFTAB
- NAMTAB
- ARGTAB



EXPANDING

ARGTAB

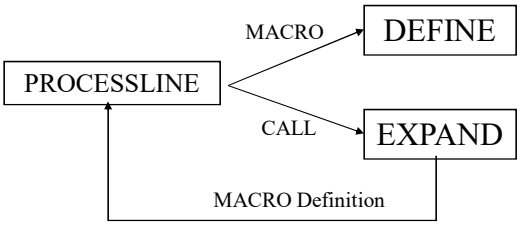
F1
buf
len

One-Pass Macro Processor

```
begin {macro processor}
    EXPANDING := FALSE
    while OPCODE ≠ 'END' do
        begin
            GETLINE
            PROCESSLINE
        end {while}
end
```

Source program
Or
Macro
definition

```
Procedure GETLINE
Begin
        if EXPANDING then
        begin
                get next line from DEFTAB
                substitute arguments from ARGTAB
        end
        else
                read next line from input file
end
```




```
procedure PROCESSLINE
    begin
        search NAMTAB for OPCODE
        if found then
           EXPAND
        else if OPCODE = 'MACRO' then
           DEFINE
        else write source line to expanded file
    end {PROCESSLINE}
```

Algorithm for a one-pass macro processor.

```
Procedure DEFINE
begin
  enter macro name in NAMETAB
  enter macro prototype in DEFTAB
  level = 1
  while level > 0 do
        GETLINE
        if this is not a comment line then
        begin
                substitute positional notation for parameter
                enter line in DEFTAB
                if opcode = 'MACRO' then
                        level = level + 1
                else if opcode = 'MEND'
                        level = level – 1
        end
        store in NAMETAB pointers to beginning and end
end
```

```
Procedure  EXPAND
Begin
        EXPANDING = true
        get first line of macro definition from DEFTAB
        setup arguments from macro invocation in ARGTAB
        write macro invocation to expand file as a comment
        GETLINE
        while not end of macro definition do
          begin
                PROCESSLINE
                GETLINE
          end
        EXPANDING = FALSE
end
```

Nested Macro Expansion

```
RDBUFF      MACRO       &BUFADR, &RECLTH, &INDEV
            CLEAR       X
            :
RDCHAR      MACRO       &IN
            :
            MEND                :
            MEND
            .
            .
            .
            RDBUFF      BUF, 1024, F1
            .
            .
            RDCHAR      F2
```

Recursive Macro Expansion

- Macro invocations within macros
  - process macro invocation during expansion time
- Recursive macro expansion
  - Example

```
RDBUFF   MACRO   &BUFADR, &RECLTH, &INDEV
         CLEAR   X
         :
         RDCHAR  &INDEV
         :
         MEND

RDCHAR   MACRO   &IN
         CLEAR   Y
         :
         MEND
```

```
RDBUFF   MACRO   &BUFADR, &RECLTH, &INDEV
         CLEAR   X
         :
         RDCHAR  &INDEV
         :
         MEND

RDCHAR   MACRO   &IN
         :
         MEND
```

Call: RDBUFF BUF, 1024, F1

- Problems:
  ARGTAB
  EXPANDING
- Solution
  Recursive call
  While loop with stack

QUIZ

```
1    copy     start   0
2    rdrec    macro   &in
7             td      =x'&in'
8             jeq     *-3
9             mend

17   first    stl     retadr
18            rdrec   f1
19            lda     length
20            comp    #0
21            rdrec   f2
23            J       @retadr
26   retadr   resw    1
27   length   resw    1
29            end     first
```

Show entries in NAMTAB and DEFTAB and output of macro processor