```
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMM MMM MMM MMM
SSS             UUU          UUU  MMM  MMM  MMM MMM
SSS             UUU          UUU  MMM   M   MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
```

FILEID**SUMED

```
  SSSSSSSS  UU      UU  MM      MM  EEEEEEEEEE  DDDDDDDD
  SSSSSSSS  UU      UU  MM      MM  EEEEEEEEEE  DDDDDDDD
SS          UU      UU  MMMM  MMMM  EE          DD      DD
SS          UU      UU  MMMM  MMMM  EE          DD      DD
SS          UU      UU  MM  MM  MM  EE          DD      DD
SS          UU      UU  MM  MM  MM  EE          DD      DD
  SSSSSS    UU      UU  MM      MM  EEEEEEEE    DD      DD
  SSSSSS    UU      UU  MM      MM  EEEEEEEE    DD      DD
        SS  UU      UU  MM      MM  EE          DD      DD
        SS  UU      UU  MM      MM  EE          DD      DD
        SS  UU      UU  MM      MM  EE          DD      DD      ....
        SS  UU      UU  MM      MM  EE          DD      DD      ....
SSSSSSSS    UUUUUUUUUU  MM      MM  EEEEEEEEEE  DDDDDDDD        ....
SSSSSSSS    UUUUUUUUUU  MM      MM  EEEEEEEEEE  DDDDDDDD        ....


LL          IIIIII      SSSSSSSS
LL          IIIIII      SSSSSSSS
LL            II      SS
LL            II      SS
LL            II      SS
LL            II      SS
LL            II        SSSSSS
LL            II        SSSSSS
LL            II              SS
LL            II              SS
LL            II              SS
LL            II              SS
LLLLLLLLLL  IIIIII    SSSSSSSS
LLLLLLLLLL  IIIIII    SSSSSSSS
```

```
                     0000     1 ;
                     0000     2 ; Version:      'V04-000'
                     0000     3 ;
                     0000     4 ;*****************************************************************************
                     0000     5 ;*                                                                           *
                     0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
                     0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
                     0000     8 ;*  ALL RIGHTS RESERVED.                                                     *
                     0000     9 ;*                                                                           *
                     0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
                     0000    11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
                     0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
                     0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
                     0000    14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
                     0000    15 ;*  TRANSFERRED.                                                             *
                     0000    16 ;*                                                                           *
                     0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
                     0000    18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
                     0000    19 ;*  CORPORATION.                                                             *
                     0000    20 ;*                                                                           *
                     0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
                     0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
                     0000    23 ;*                                                                           *
                     0000    24 ;*                                                                           *
                     0000    25 ;*****************************************************************************
                     0000    26 ;
                     0000    27 ;
                     0000    28 ; Assembly parameters
                     0000    29 ;
00000200             0000    30         BUF_SIZE = 512                  ; Size in bytes of slipr input buffers
00000084             0000    31         CMD_SIZE = 132                  ; Size of input command line
                     0000    32 ;
                     0000    33         $NAMDEF
                     0000    34         $RABDEF
                     0000    35         $FABDEF
                     0000    36         $CLIDEF
                     0000    37 ;
                     0000    38 ; Edit node offsets
                     0000    39 ;
00000000             0000    40         ED$L_FWD    = 0                 ; Forward pointer
00000004             0000    41         ED$L_BWD    = 4                 ; Backword pointer
00000008             0000    42         ED$W_LOC1   = 8                 ; Locator 1
0000000A             0000    43         ED$W_LOC2   = 10                ; Locator 2
0000000C             0000    44         ED$W_LINES  = 12                ; Insert lines
0000000E             0000    45         ED$W_RFA    = 14                ; Record file address (3 words)
00000014             0000    46         ED$L_FILE   = 20                ; File node pointer
00000018             0000    47         ED$B_FLAGS  = 24                ; Flags
00000019             0000    48         ED$B_FILENO= 25                 ; File number
                     0000    49 ;
0000001A             0000    50         ED$K_BLN    = 26
                     0000    51 ;
                     0000    52 ;
                     0000    53 ; File node offsets
                     0000    54 ;
00000000             0000    55         SLP$L_FWD   = 0                 ; Forward pointer
00000004             0000    56         SLP$L_BWD   = 4                 ; Backward pointer
00000008             0000    57         SLP$W_LOC1  = 8                 ; locator-1
```

```
0000000A  0000     58          SLP$W_LOC2 = 10                 ; Locator-2
0000000C  0000     59          SLP$B_FLAGS= 12                 ; Flags
0000000D  0000     60          SLP$B_FILENO = 13               ; File priority
0000000E  0000     61          SLP$W_DOT  = 14                 ; Dot value
00000010  0000     62          SLP$Q_AUDDS= 16                 ; Audit string desciptor
00000018  0000     63          SLP$T_AUDST= 24                 ; Audit string
00000028  0000     64          SLP$Q_AUCDS= 40                 ; Current audit string descriptor
00000030  0000     65          SLP$T_AUCST= 48                 ; Current audit string
00000040  0000     66          SLP$Q_CMNT = 64                 ; Comment descriptor
00000048  0000     67          SLP$T_NAM  = 72                 ; NAM block
          0000     68 ;
000000A8  0000     69          SLP$K_BLN  = SLP$T_NAM + NAM$K_BLN
          0000     70 ;
          0000     71 ;
          0000     72 ; Macro to print error message
          0000     73 ;
          0000     74          .MACRO  ERRMSG  NAME,LIST
          0000     75          $$ = 0
          0000     76          .IRP    L,<LIST>
          0000     77          PUSHL   L
          0000     78          $$=$$+1
          0000     79          .ENDR
          0000     80          PUSHL   #$$
          0000     81          MOVL    #MER$_'NAME',R0
          0000     82          PUSHL   R0
          0000     83          CALLS   #$$+2,G^LIB$SIGNAL
          0000     84          .ENDM   ERRMSG
```

```
                     1          .TITLE  SUMED
                     2          .IDENT  /V04-000/
                     3
                     4 ;
                     5 ;**********************************************************************
                     6 ;*                                                                    *
                     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
                     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
                     9 ;*  ALL RIGHTS RESERVED.                                              *
                    10 ;*                                                                    *
                    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
                    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
                    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
                    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
                    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
                    16 ;*  TRANSFERRED.                                                      *
                    17 ;*                                                                    *
                    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
                    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
                    20 ;*  CORPORATION.                                                      *
                    21 ;*                                                                    *
                    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
                    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.           *
                    24 ;*                                                                    *
                    25 ;*                                                                    *
                    26 ;**********************************************************************
                    27 ;
                    28 ; Procedure to merge slipr files - Main loop
                    29 ;
                    30 ;
                    31 ; Each input file is read and for every editing operation an
                    32 ; Edit node is created.  This node is linked into the list of editing
                    33 ; nodes in locator-1 sequence.  If there is an existing node with
                    34 ; the same locator-1 the new node will be linked in front of it.  This
                    35 ; gives it precedence to an earlier edit.  As each node is linked checks
                    36 ; for clashes are made.  If a clash is detected an error message is
                    37 ; printed.
                    38 ;
                    39 ; When all files have been read the complete Edit nodes list is processed
                    40 ; to form the merged output file
                    41 ;
                    42          $RABDEF
                    43
                    44          .PSECT  $CODE,EXE,NOWRT
                    45 ;
                    46 ;
                    47 ;
                    48 ;
                    49 MERGE_FILES::
          0000      50          .WORD   0
       07 10  0002  51          BSB     PASS_ONE
    03 50 E9  0004  52          BLBC    R0,10$
     03EF 30  0007  53          BSBW    PASS_TWO
              000A  54 10$:
           04 000A  55          RET
```

```
                        000B     57 ;
                        000B     58 ; Subroutine to read and process all input files
                        000B     59 ;
                        000B     60 ; Inputs:
                        000B     61 ;       Files list
                        000B     62 ;
                        000B     63 ; Outputs:
                        000B     64 ;       Edit nodes
                        000B     65 ;
                        000B     66 ;
                        000B     67 PASS_ONE:
    5A   0000'CF   DE   000B     68          MOVAL   W^FILE_NODES,R10        ; Get address of file nodes
                        0010     69 10$:
        5A   6A    D0   0010     70          MOVL    (R10),R10               ; Get next node
00000000'8F   5A    D1   0013     71          CMPL    R10,#FILE_NODES         ; At end of list?
           03    12   001A     72          BNEQ    5$                      ; No if NEQ
         00CF    31   001C     73          BRW     80$
                        001F     74 5$:
        53   5A    D0   001F     75          MOVL    R10,R3                  ; Form NAM block pointer
53  00000048 8F    C0   0022     76          ADDL2   #SLP$T_NAM,R3
    54   0000'CF   DE   0029     77          MOVAL   W^INPUT_FAB,R4          ; and FAB pointer
                        002E     78          $FAB_STORE FAB = R4, NAM = (R3)
      58   04 A3   D0   0032     79          MOVL    NAM$L_RSA(R3),R8        ; Get file name in case of error
      59   03 A3   9A   0036     80          MOVZBL  NAM$B_RSL(R3),R9
                        003A     81          $OPEN   FAB = R4
           1E 50    E8   0043     82          BLBS    R0,20$                  ; OK if LBS
           0C A4    DD   0046     83          PUSHL   FAB$L_STV(R4)
              50    DD   0049     84          PUSHL   R0
                        004B     85          ERRMSG  REOPEN,<R8,R9>
            0083    31   0061     86          BRW     66$
                        0064     87 20$:
    52   0000'CF   DE   0064     88          MOVAL   W^INPUT_RAB,R2
                        0069     89          $CONNECT RAB = R2
           1D 50    E8   0072     90          BLBS    R0,30$                  ; OK if LBS
           0C A2    DD   0075     91          PUSHL   RAB$L_STV(R2)
              50    DD   0078     92          PUSHL   R0
                        007A     93          ERRMSG  CONNEC,<R8,R9>
              55    11   0090     94          BRB     66$
                        0092     95 30$:
              5B    10   0092     96          BSB     SET_UP_NODES
                        0094     97          $DISCONNECT RAB = R2
           1D 50    E8   009D     98          BLBS    R0,40$                  ; OK if LBS
           0C A2    DD   00A0     99          PUSHL   RAB$L_STV(R2)
              50    DD   00A3    100          PUSHL   R0
                        00A5    101          ERRMSG  DISCON,<R8,R9>
              2A    11   00BB    102          BRB     66$
                        00BD    103 40$:
                        00BD    104          $CLOSE  FAB = R4
           03 50    E9   00C6    105          BLBC    R0,50$                  ; Error if LBC
            FF44    31   00C9    106          BRW     10$
                        00CC    107 50$:
           0C A2    DD   00CC    108          PUSHL   FAB$L_STV(R2)
              50    DD   00CF    109          PUSHL   R0
                        00D1    110          ERRMSG  CLOSER,<R8,R9>
                        00E7    111 66$:
00000000'GF   02    FB   00E7    112          CALLS   #2,G^LIB$SIGNAL         ; Signal error
                        00EE    113 80$:
```

```
                05  00EE   114          RSB
```

```
                        00EF    116          .SBTTL  SET_UP_NODES
                        00EF    117 ;
                        00EF    118 ; Subroutine to form all edit_nodes
                        00EF    119 ;
                        00EF    120 ; Inputs:
                        00EF    121 ;       R2  = RAB address
                        00EF    122 ;       R8  = file spec address
                        00EF    123 ;       R9  = file spec size
                        00EF    124 ;       R10 = file node address
                        00EF    125 ;
                        00EF    126 ; Outputs:
                        00EF    127 ;       R0  = Success/error status
                        00EF    128 ;
                        00EF    129 ;
                        00EF    130 SET_UP_NODES:
    00DC 8F     BB      00EF    131          PUSHR   #^M<R2,R3,R4,R6,R7>
10 A2    01     D0      00F3    132          MOVL    #1,RAB$W_RFA+0(R2)      ; Initialise record file address (3 words)
   14 A2        B4      00F7    133          CLRW    RAB$W_RFA+4(R2)
                        00FA    134 10$:
        7F      10      00FA    135          BSB     GET_NODE                ; Get empty node
     77 50      E9      00FC    136          BLBC    R0,60$                  ; Error if LBC
14 A5    5A     D0      00FF    137          MOVL    R10,ED$L_FILE(R5)       ; File address
19 A5    0D AA  90      0103    138          MOVB    SLP$B_FILENO(R10), -    ; File number
                        0108    139                  ED$B_FILENO(R5)
0E A5    10 A2  D0      0108    140          MOVL    RAB$W_RFA+0(R2),ED$W_RFA+0(R5) ; Record file adress (3 words)
12 A5    14 A2  B0      010D    141          MOVW    RAB$W_RFA+4(R2),ED$W_RFA+4(R5)
         0C A5  B4      0112    142          CLRW    ED$W_LINES(R5)
                        0115    143 20$:
08 A5    08 AA  B0      0115    144          MOVW    SLP$W_LOC1(R10),ED$W_LOC1(R5) ; Locator numbers
0A A5    0A AA  B0      011A    145          MOVW    SLP$W_LOC2(R10),ED$W_LOC2(R5)
18 A5    0C AA  90      011F    146          MOVB    SLP$B_FLAGS(R10),ED$B_FLAGS(R5) ; and flags byte
                        0124    147 30$:
      027D      30      0124    148          BSBW    READ_LINE               ; Read line from input file
     4C 50      E9      0127    149          BLBC    R0,60$                  ; Error if LBC, process as end of edit
      05F4      30      012A    150          BSBW    COMMAND_CHECK           ; Check for command
     18 50      E8      012D    151          BLBS    R0,35$                  ; OK if LBS
                        0130    152          ERRMSG  SYNTAX,<R6,R7>          ; SLP command syntax error
         DC     11      0146    153          BRB     30$                     ; Ignore line
                        0148    154 35$:
   54    02     D3      0148    155          BITL    #2,R4                   ; Data terminating command?
         0A     12      014B    156          BNEQ    40$                     ; Yes if NEQ
   54    01     93      014D    157          BITB    #1,R4                   ; Data or normal command?
         D2     12      0150    158          BNEQ    30$                     ; Normal command if NEQ
      0C A5     B6      0152    159          INCW    ED$W_LINES(R5)          ; Increment number of insert lines for
         CD     11      0155    160          BRB     30$                     ; this edit
                        0157    161 40$:
      08 A5     B5      0157    162          TSTW    ED$W_LOC1(R5)           ; If Loc-1 and Loc-2 = 0 and Lines <> 0
         13     12      015A    163          BNEQ    50$                     ; there is an insert in front of
      0A A5     B5      015C    164          TSTW    ED$W_LOC2(R5)           ; the file, otherwise throw this
         0E     12      015F    165          BNEQ    50$                     ; Edit node away
      0C A5     B5      0161    166          TSTW    ED$W_LINES(R5)
         09     12      0164    167          BNEQ    50$
   54    04     D3      0166    168          BITL    #4,R4                   ; Was command "/"?
         04     13      0169    169          BEQL    50$                     ; No if EQL
         26     10      016B    170          BSB     RETURN_NODE             ; Return node
         07     11      016D    171          BRB     60$
                        016F    172 50$:
```

```
                37  10  016F    173             BSB     INSERT_NODE                     ; Insert node
54              04  D3  0171    174             BITL    #4,R4                           ; Was last command "/"?
                84  13  0174    175             BEQL    10$                             ; No if EQL, back for next edit
                        0176    176 60$:
00DC 8F         BA  0176    177                 POPR    #^M<R2,R3,R4,R6,R7>
                05  017A    178                 RSB
```

```
                     017B  180          .SBTTL  GET_NODE
                     017B  181 ;
                     017B  182 ; Subroutine to get edit node
                     017B  183 ;
                     017B  184 ; Inputs:
                     017B  185 ;       none
                     017B  186 ;
                     017B  187 ; Outputs:
                     017B  188 ;       R0  = Success/error status
                     017B  189 ;       R5  = Node address
                     017B  190 ;
                     017B  191 ;
                     017B  192 GET_NODE:
          0000'CF    DF  017B  193          PUSHAL  W^VIRT_ADDR             ; Address to return VA
          0000'CF    DF  017F  194          PUSHAL  W^ED_SIZE               ; Size of edit node
00000000'GF    02    FB  0183  195          CALLS   #2,G^LIB$GET_VM         ; Get memory
        05 50        E9  018A  196          BLBC    R0,10$                  ; Error if LBC
    55  0000'CF      D0  018D  197          MOVL    W^VIRT_ADDR,R5          ; Put node address in R5
                     0192  198 10$:
                     05  0192  199          RSB
```

```
                     0193   201          .SBTTL  RETURN_NODE
                     0193   202 ;
                     0193   203 ; Subroutine to return edit node
                     0193   204 ;
                     0193   205 ; Inputs:
                     0193   206 ;       R5 = Node address
                     0193   207 ;
                     0193   208 ; Outputs:
                     0193   209 ;       R0 = Success/error status
                     0193   210 ;
                     0193   211 ;
                     0193   212 RETURN_NODE:
      0000'CF    55  DO  0193   213          MOVL    R5,W^VIRT_ADDR          ; Store address of node
           0000'CF  DF  0198   214          PUSHAL  W^VIRT_ADDR             ; Push parameters
           0000'CF  DF  019C   215          PUSHAL  W^ED_SIZE
00000000'GF    02  FB  01A0   216          CALLS   #2,G^LIB$FREE_VM
                   05  01A7   217          RSB
```

```
                        01A8   219          .SBTTL  INSERT_NODE
                        01A8   220 ;
                        01A8   221 ; Subroutine to insert node into edit list and report any clashes
                        01A8   222 ; with previous edits
                        01A8   223 ;
                        01A8   224 ; Inputs:
                        01A8   225 ;       R5  = address of node to insert
                        01A8   226 ;
                        01A8   227 ; Outputs:
                        01A8   228 ;       None
                        01A8   229 ;
                        01A8   230 ;
                        01A8   231 INSERT_NODE:
                  1C    BB    01A8   232          PUSHR   #^M<R2,R3,R4>
      53    0000'CF     DE    01AA   233          MOVAL   W^EDIT_NODES,R3         ; Get address of list head
                  52    D4    01AF   234          CLRL    R2                      ; Initialise clash count
                        01B1   235 10$:
                  53    63    D0    01B1   236          MOVL    (R3),R3                 ; Get next node
00000000'8F       53    D1    01B4   237          CMPL    R3,#EDIT_NODES          ; At end of list?
                  12    13    01BB   238          BEQL    20$                     ; Yes if EQL
   08 A3    08 A5    B1    01BD   239          CMPW    ED$W_LOC1(R5),ED$W_LOC1(R3) ; Is new LOC-1 <= current LOC-1
                  0B    15    01C2   240          BLEQ    20$                     ; Yes if LEQ, found place for node
   0A A3    08 A5    B1    01C4   241          CMPW    ED$W_LOC1(R5),ED$W_LOC2(R3) ; Does current LOC-2 overlap with new LO
                  E6    14    01C9   242          BGTR    10$                     ; No if GTR
                  29    10    01CB   243          BSB     REPORT_CLASH
                  E2    11    01CD   244          BRB     10$
                        01CF   245 20$:
   04 B3    65    0E    01CF   246          INSQUE  (R5),@ED$L_BWD(R3)      ; Insert new node into list
   54    0A A5    3C    01D3   247          MOVZWL  ED$W_LOC2(R5),R4        ; Get LOC-2 value
                  04    12    01D7   248          BNEQ    30$
   54    08 A5    3C    01D9   249          MOVZWL  ED$W_LOC1(R5),R4
                        01DD   250 30$:
00000000'8F       53    D1    01DD   251          CMPL    R3,#EDIT_NODES          ; End of list?
                  0D    13    01E4   252          BEQL    40$                     ; Yes if EQL
   08 A3    54    B1    01E6   253          CMPW    R4,ED$W_LOC1(R3)        ; Does edit clash?
                  07    19    01EA   254          BLSS    40$                     ; No if LSS
                  08    10    01EC   255          BSB     REPORT_CLASH
      53    63    D0    01EE   256          MOVL    (R3),R3                 ; Advance to next node
                  EA    11    01F1   257          BRB     30$
                        01F3   258 40$:
                  1C    BA    01F3   259          POPR    #^M<R2,R3,R4>
                  05    01F5   260          RSB
```

```
                                01F6   262          .SBTTL  REPORT_CLASH
                                01F6   263 ;
                                01F6   264 ; Subroutine to report clashes between edits
                                01F6   265 ;
                                01F6   266 ; Inputs:
                                01F6   267 ;       R2  = Number of clashes
                                01F6   268 ;       R3  = Node address of clashing edit
                                01F6   269 ;       R5  = Node address of new edit
                                01F6   270 ;
                                01F6   271 ; Outputs:
                                01F6   272 ;       R2  = Number of clashes
                                01F6   273 ;
                                01F6   274 ;
                                01F6   275 REPORT_CLASH:
                    08 A3   B5  01F6   276          TSTW    ED$W_LOC1(R3)           ; If LOC-1, LOC-2 and LINES all
                       0A   12  01F9   277          BNEQ    5$                      ; equal zero node is not full edit
                    0A A3   B5  01FB   278          TSTW    ED$W_LOC2(R3)
                       05   12  01FE   279          BNEQ    5$
                    0C A3   B5  0200   280          TSTW    ED$W_LINES(R3)
                       36   13  0203   281          BEQL    20$
                                0205   282 5$:
          09 18 A3  00000000'8F  E1  0205   283          BBC     #SLPV_SUPPRESS,-        ; Branch if clash report has
                                020E   284                  ED$B_FLAGS(R3),7$       ; not been suppressed
          24 18 A5  00000000'8F  E0  020E   285          BBS     #SLPV_SUPPRESS,-        ; Branch if clash report has
                                0217   286                  ED$B_FLAGS(R5),20$      ; been suppressed
                                0217   287 7$:
                          52    D5  0217   288          TSTL    R2                      ; Is this first clash?
                          1C    12  0219   289          BNEQ    10$                     ; No if NEQ
                          53    DD  021B   290          PUSHL   R3                      ; Save R3
                    53    55    D0  021D   291          MOVL    R5,R3
                                0220   292          ERRMSG  CLASH                   ; Report new clash
                       08     10  0232   293          BSB     CLASH_LINE              ; Print edit line
                       53  8ED0   0234   294          POPL    R3                      ; Restore R3
                                0237   295 10$:
                       52     D6  0237   296          INCL    R2                      ; Increment number of clashes
                       01     10  0239   297          BSB     CLASH_LINE              ; Print edit line
                                023B   298 20$:
                              05  023B   299          RSB
```

```
                                023C    301          .SBTTL  CLASH_LINE
                                023C    302 ;
                                023C    303 ; Subroutine to print random line from slp input file
                                023C    304 ;
                                023C    305 ;
                                023C    306 ; Inputs:
                                023C    307 ;       R3 = Edit node address
                                023C    308 ;
                                023C    309 ; Outputs:
                                023C    310 ;       None
                                023C    311 ;
                                023C    312 ;
                                023C    313 CLASH_LINE:
                        07FC 8F   BB    023C    314          PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
                             6B   10    0240    315          BSB     RANDOM_OPEN                     ; Open file for random access
                           5C 50  E9    0242    316          BLBC    R0,70$                          ; Error if LBC
                     52    0000'CF  DE    0245    317          MOVAL   W^RANDOM_RAB,R2         ; Set pointer to RAB
                   10 A2     0E A3  D0    024A    318          MOVL    ED$W_RFA+0(R3),RAB$W_RFA+0(R2) ; Copy record file address
                   14 A2     12 A3  B0    024F    319          MOVW    ED$W_RFA+4(R3),RAB$W_RFA+4(R2) ; (3 words)
                                    0254    320          $GET    RAB = R2
                             1D 50  E8    025D    321          BLBS    R0,60$
                             0C A2  DD    0260    322          PUSHL   RAB$L_STV(R2)
                                50  DD    0263    323          PUSHL   R0
                                    0265    324          ERRMSG  READER,<R8,R9>
                                24  11    027B    325          BRB     70$
                                    027D    326 60$:
                           56   28 A2  D0    027D    327          MOVL    RAB$L_RBF(R2),R6
                           57   22 A2  3C    0281    328          MOVZWL  RAB$W_RSZ(R2),R7
                                    0285    329          ERRMSG  CLSHLN,<R6,R7,R8,R9>
                                07  11    029F    330          BRB     80$
                                    02A1    331 70$:
                00000000'GF     02  FB    02A1    332          CALLS   #2,G^LIB$SIGNAL
                                    02A8    333 80$:
                        07FC 8F   BA    02A8    334          POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
                                05  02AC    335          RSB
```

```
                     02AD    337          .SBTTL  RANDOM_OPEN
                     02AD    338 ;
                     02AD    339 ; Subroutine to open a slp input file for random access.  If the
                     02AD    340 ; file is the currently open file then no file access are performed.
                     02AD    341 ; Otherwise, the current file is close and disonnected from the RAB
                     02AD    342 ; and the rqueisted file opened and connected to the RAB
                     02AD    343 ;
                     02AD    344 ;
                     02AD    345 ; Inputs:
                     02AD    346 ;       R3 = Edit node address
                     02AD    347 ;
                     02AD    348 ; Outputs:
                     02AD    349 ;       R8 = Address of current file specification
                     02AD    350 ;       R9 = Size of current file specification
                     02AD    351 ;
                     02AD    352 ;
                     02AD    353 RANDOM_OPEN:
        04FC 8F      BB     02AD    354          PUSHR   #^M<R2,R3,R4,R5,R6,R7,R10>
        50    01     D0     02B1    355          MOVL    #1,R0                   ; Assume success
   5A    14 A3     D0     02B4    356          MOVL    ED$L_FILE(R3),R10       ; Get file node address
52    0000'CF     DE     02B8    357          MOVAL   W^RANDOM_RAB,R2         ; Set RAB pointer
54    0000'CF     DE     02BD    358          MOVAL   W^RANDOM_FAB,R4         ; FAB pointer
      55    5A     D0     02C2    359          MOVL    R10,R5                  ; NAM pointer
55    00000048 8F      C0     02C5    360          ADDL2   #SLP$T_NAM,R5
   58    04 A5     D0     02CC    361          MOVL    NAM$L_RSA(R5),R8        ; Pointers to file spec
   59    03 A5     9A     02D0    362          MOVZBL  NAM$B_RSL(R5),R9        ; string and size
      0000'CF     D5     02D4    363          TSTL    W^RANDOM_FILE           ; Is a file open?
            61     13     02D8    364          BEQL    20$                     ; No if EQL
   0000'CF    5A     D1     02DA    365          CMPL    R10,W^RANDOM_FILE       ; Is it file required?
            03     12     02DF    366          BNEQ    5$                      ; No if NEQ
         00BB    31     02E1    367          BRW     80$
                     02E4    368 5$:
(J00'CF     D4     02E4    369          CLRL    W^RANDOM_FILE           ; Close current file
                     02E8    370          $DISCONNECT RAB = R2
   1E 50     E8     02F1    371          BLBS    R0,10$
   0C A2     DD     02F4    372          PUSHL   RAB$L_STV(R2)
      50     DD     02F7    373          PUSHL   R0
                     02F9    374          ERRMSG  DISCON,<R8,R9>
      0086    31     030F    375          BRW     70$
                     0312    376 10$:
                     0312    377          $CLOSE  FAB = R4
   1D 50     E8     031B    378          BLBS    R0,20$
   0C A4     DD     031E    379          PUSHL   FAB$L_STV(R4)
      50     DD     0321    380          PUSHL   R0
                     0323    381          ERRMSG  CLOSER,<R8,R9>
      5D     11     0339    382          BRB     70$
                     033B    383 20$:
                     033B    384          $FAB_STORE FAB = R4, NAM = (R5)
                     033F    385          $OPEN   FAB = R4
   1D 50     E8     0348    386          BLBS    R0,30$
   0C A4     DD     034B    387          PUSHL   FAB$L_STV(R4)
      50     DD     034E    388          PUSHL   R0
                     0350    389          ERRMSG  REOPEN,<R8,R9>
      30     11     0366    390          BRB     70$
                     0368    391 30$:
                     0368    392          $CONNECT RAB = R2
   1D 50     E8     0371    393          BLBS    R0,40$
```

RANDOM_OPEN

```
                OC A2   DD  0374   394          PUSHL   RAB$L_STV(R2)
                   50   DD  0377   395          PUSHL   R0
                                   396          ERRMSG  CONNEC,<R8,R9>
                   07   11  038F   397          BRB     70$
                                   398 40$:
          0000'CF  5A   D0  0391   399          MOVL    R10,W^RANDOM_FILE               ; Set new file open
                   07   11  0396   400          BRB     80$
                                   401 70$:
00000000'GF        02   FB  0398   402          CALLS   #2,G^LIB$SIGNAL
                                   403 80$:
              04FC 8F   BA  039F   404          POPR    #^M<R2,R3,R4,R5,R6,R7,R10>
                        05  03A3   405          RSB
```

```
                        03A4    407          .SBTTL  READ_LINE
                        03A4    408 ;
                        03A4    409 ; Subroutine to read line sequentially from current slipr input file
                        03A4    410 ;
                        03A4    411 ; Inputs:
                        03A4    412 ;       R8 = Address of file spec
                        03A4    413 ;       R9 = Size of file spec
                        03A4    414 ;
                        03A4    415 ; Outputs:
                        03A4    416 ;       R0 = success/error status
                        03A4    417 ;       R6 = Line buffer address
                        03A4    418 ;       R7 = Line size
                        03A4    419 ;
                        03A4    420 ;
                        03A4    421 READ_LINE:
            52   DD     03A4    422          PUSHL   R2
    52  0000'CF   DE     03A6    423          MOVAL   W^INPUT_RAB,R2          ; Get RAB address
                        03AB    424          $GET    RAB = R2                ; Read line
          36 50   E8     03B4    425          BLBS    R0,20$                  ; OK if LBS
00000000'8F   50   D1     03B7    426          CMPL    R0,#RMS$_EOF            ; Unexpected EOF?
              1F   12     03BE    427          BNEQ    10$                     ; No if NEQ
                        03C0    428          ERRMSG  INPEOF,<R8,R9>          ; Report error
50  00000000'8F   D0     03D6    429          MOVL    #RMS$_EOF,R0            ; Reset R0 to indicate error
              16   11     03DD    430          BRB     30$
                        03DF    431 10$:
          0C A2   DD     03DF    432          PUSHL   RAB$L_STV(R2)           ; Signal error
              50   DD     03E2    433          PUSHL   R0
00000000'GF   02   FB     03E4    434          CALLS   #2,G^LIB$SIGNAL
              08   11     03EB    435          BRB     30$
                        03ED    436 20$:
    56  28 A2   D0     03ED    437          MOVL    RAB$L_RBF(R2),R6        ; Set buffer address
    57  22 A2   3C     03F1    438          MOVZWL  RAB$W_RSZ(R2),R7        ; and line size
                        03F5    439 30$:
        52 8ED0     03F5    440          POPL    R2
              05    03F8    441          RSB
```

```
                        03F9   443          .SBTTL  PASS_TWO
                        03F9   444 ;
                        03F9   445 ; Subroutine to perform Pass 2
                        03F9   446 ;
                        03F9   447 ; This routine processes each edit node, by forming and outputing the
                        03F9   448 ; editing command and then copying any data lines.  For each edit it
                        03F9   449 ; checks that the current audit trail switch (on or off) and audit
                        03F9   450 ; trail string are what are required for that file.  If not commands
                        03F9   451 ; are generated and output to set the audit trail switch and string
                        03F9   452 ; to the correct values
                        03F9   453 ;
                        03F9   454 ;
                        03F9   455 ; Inputs:
                        03F9   456 ;       Edit nodes
                        03F9   457 ;
                        03F9   458 ; Outputs:
                        03F9   459 ;       None
                        03F9   460 ;
                        03F9   461 ; Define R2 flag bits
                        03F9   462 ;
                        03F9   463 _VIELD  SUM,0,< -
                        03F9   464         <EXPED,,M> -                    ; Expected edit command has been seen
                        03F9   465         <1STEDIT,,M> -                  ; 1st edit has been processed
                        03F9   466         <SUPPRESS,,M> -                 ; Edit is '--' form
                        03F9   467         <SPRCDE,,M> -                   ; Insert/replace lines being superceded
                        03F9   468         >
                        03F9   469 ;
                        03F9   470 ;
                        03F9   471 PASS_TWO:
          53  0000'CF   DE 03F9   472         MOVAL   W^EDIT_NODES,R3         ; Set up edit nodes pointer
                        03FE   473 10$:
          53    63    D0 03FE   474         MOVL    (R3),R3                 ; Get next node
00000000'8F     53    D1 0401   475         CMPL    R3,#EDIT_NODES          ; End of list?
              0B    13 0408   476         BEQL    20$                     ; Yes if EQL
              52    D4 040A   477         CLRL    R2                      ; Clear all flags
            00AD    30 040C   478         BSBW    RANGE_EDIT              ; Find range of next edit
            000F    30 040F   479         BSBW    PROCESS_EDIT            ; Process next edit
           E9 50    E8 0412   480         BLBS    R0,10$                  ; All ok if LBS
                        0415   481 20$:
      56  0000'CF   DE 0415   482         MOVAL   W^END_EDIT,R6           ; Get address of end edit line
          57  01    D0 041A   483         MOVL    #1,R7                   ; and set its size
            02BB    30 041D   484         BSBW    WRITE_LINE              ; Write line to output file
                    05 0420   485         RSB
```

```
                     0421   487          .SBTTL  PROCESS_EDIT
                     0421   488 ;
                     0421   489 ; Subroutine to process next edit operation
                     0421   490 ;
                     0421   491 ; Inputs:
                     0421   492 ;
                     0421   493 ;       R2  = Flags
                     0421   494 ;       R3  = First edit node address
                     0421   495 ;       R5  = Last edit node address
                     0421   496 ;       R11 = Highest loc-2 of edit
                     0421   497 ;
                     0421   498 ; Outputs:
                     0421   499 ;
                     0421   500 ;       R3  = Last edit node address
                     0421   501
                     0421   502 ;
                     0421   503 ;
                     0421   504 PROCESS_EDIT:
          53   DD    0421   505          PUSHL   R3                      ; Save first node address
                     0423   506 5$:
     5A   14 A3   D0  0423   507          MOVL    ED$L_FILE(R3),R10       ; Get file node address
           00DE   30  0427   508          BSBW    ACCESS_EDIT             ; Set up RMS to read edit
        03 50   E8  042A   509          BLBS    R0,7$                   ; OK if LBS
         0088   31  042D   510          BRW     120$                    ; Error
                     0430   511 7$:
        08 A3   B5  0430   512          TSTW    ED$W_LOC1(R3)           ; Is loc-1 zero?
           03   12  0433   513          BNEQ    10$                     ; No if NEQ
     52    01   C8  0435   514          BISL2   #SUM_M_EXPED,R2         ; Set expected edit seen flag
                     0438   515 10$:
           00EC   30  0438   516          BSBW    READ_RECORD             ; Get next record
        7A 50   E9  043B   517          BLBC    R0,120$                 ; Error if LBC
         02E0   30  043E   518          BSBW    COMMAND_CHECK           ; Check for command
        F4 50   E9  0441   519          BLBC    R0,10$                  ; Error if LBC
  12 54    00   E1  0444   520          BBC     #0,R4,30$               ; Branch if data line
  17 54    01   E0  0448   521          BBS     #1,R4,40$               ; Branch if edit terminating command
                     044C   522 15$:
        10 AA   B5  044C   523          TSTW    SLP$Q_AUDDS(R10)        ; Was new audit trail specified?
           03   13  044F   524          BEQL    20$                     ; No if EQL
         0114   30  0451   525          BSBW    RESET_AUDIT             ; Set new current audit string
                     0454   526 20$:
0000'CF    0C AA   9B  0454   527          MOVZBW  SLP$B_FLAGS(R10),W^SLP_FLAGS ; Set new current flags
                     045A   528 30$:
     DA 52   03   E0  045A   529          BBS     #SUM_V_SPRCDE,R2,10$    ; Branch if edit being superceded
           027A   30  045E   530          BSBW    WRITE_LINE              ; Write line to output file
             D5   11  0461   531          BRB     10$                     ; Back for next node
                     0463   532 40$:
     1F 52   00   E2  0463   533          BBSS    #SUM_V_EXPED,R2,70$     ; Is this expected edit command?
     E1 52   03   E0  0467   534          BBS     #SUM_V_SPRCDE,R2,15$    ; Branch if edit is being superceded
     06 52   01   E2  046B   535          BBSS    #SUM_V_1STEDIT,R2,50$   ; Is this first edit of range?
     0A AA   5B   B0  046F   536          MOVW    R11,SLP$W_LOC2(R10)     ; Yes: set Loc-2 to highest in range
             09   11  0473   537          BRB     60$
                     0475   538 50$:
        08 AA   B4  0475   539          CLRW    SLP$W_LOC1(R10)         ; Zero both Loc-1
        0A AA   B4  0478   540          CLRW    SLP$W_LOC2(R10)         ; and Loc-2
     52    04   CA  047B   541          BICL2   #SUM_M_SUPPRESS,R2      ; Only first edit may be '--' form
                     047E   542 60$:
           010C   30  047E   543          BSBW    WRITE_EDIT              ; Write edit line
```

PROCESS_EDIT

```
                B4 50    E8  0481    544          BLBS    R0,10$                  ; OK if LBS
                   32    11  0484    545          BRB     120$
                                     546 70$:
             52    09    CA  0486    547          BICL2   #SUM_M_EXPED!SUM_M_SPRCDE,R2 ; Clear flags
             55    53    D1  0489    548          CMPL    R3,R5                   ; Last edit in range?
                   2A    13  048C    549          BEQL    120$                    ; Yes if EQL
             54    53    D0  048E    550          MOVL    R3,R4                   ; Set temp node pointer
             53    63    D0  0491    551          MOVL    (R3),R3                 ; Move to next node
                                     552 80$:
          19 A3    19 A4  91  0494    553          CMPB    ED$B_FILENO(R4), -      ; Is file number of this edit =>
                                0499    554                  ED$B_FILENO(R3)     ; file number of next edit?
                   0C    19  0499    555          BLSS    90$                     ; No if LSS, inserts are not superceded
                08 A4    B5  049B    556          TSTW    ED$W_LOC1(R4)           ; Insert to front of file?
                   07    13  049E    557          BEQL    90$                     ; Yes if EQL, insert is not superceded
          08 A3    0A A4  B1  04A0    558          CMPW    ED$W_LOC2(R4), -        ; Does edit's loc-2 reach next's loc-1
                                04A5    559                  ED$W_LOC1(R3)
                   0B    18  04A5    560          BGEQ    100$                    ; Yes if GEQ, inserts are superceded
                                04A7    561 90$:
             6E    54    D1  04A7    562          CMPL    R4,(SP)                 ; Back to first node?
                   09    13  04AA    563          BEQL    110$                    ; Yes if EQL
          54    04 A4    D0  04AC    564          MOVL    ED$L_BWD(R4),R4         ; Go back to previous node
                   E2    11  04B0    565          BRB     80$                     ; and test again
                                04B2    566 100$:
             52    08    88  04B2    567          BISB    #SUM_M_SPRCDE,R2        ; Set inserts to be superceded flags
                                04B5    568 110$:
                FF6B    31  04B5    569          BRW     5$                      ; Process next edit
                                04B8    570 120$:
             5E    04    C0  04B8    571          ADDL2   #4,SP                   ; Clear stack
                         05  04BB    572          RSB
```

```
                                04BC    574          .SBTTL  RANGE_EDIT
                                04BC    575 ;
                                04BC    576 ; Subroutine to find range of next edit operation
                                04BC    577 ;
                                04BC    578 ; Inputs:
                                04BC    579 ;
                                04BC    580 ;       R2 = Flags
                                04BC    581 ;       R3 = Current (first) node
                                04BC    582 ;
                                04BC    583 ; Outputs:
                                04BC    584 ;
                                04BC    585 ;       R2 = Flags
                                04BC    586 ;       R5 = Last node of edit
                                04BC    587 ;       R11= Highest locator-2 number of edit range
                                04BC    588 ;
                                04BC    589 ;
                                04BC    590 RANGE_EDIT:
                          53    DD  04BC    591          PUSHL   R3                      ; Save R3
                       52   04    C8  04BE    592          BISL2   #SUM_M_SUPPRESS,R2      ; Assume all clash messages suppressed
                    5B    0A A3    3C  04C1    593          MOVZWL  ED$W_LOC2(R3),R11       ; Initialise highest locator-2
                       55   53    D0  04C5    594          MOVL    R3,R5                   ; and current node pointer
                                04C8    595 10$:
          03 18 A3    00000000'8F    E0  04C8    596          BBS     #SLPV_SUPPRESS, -       ; Branch if '--' form of edit
                                04D1    597                  ED$B_FLAGS(R3),15$
                       52   04    CA  04D1    598          BICL2   #SUM_M_SUPPRESS,R2      ; Edit will be '-' form
                                04D4    599 15$:
                       53   65    D0  04D4    600          MOVL    (R5),R3                 ; Point to next edit node
                00000000'8F    53    D1  04D7    601          CMPL    R3,#EDIT_NODES          ; Is it end of list?
                          24    13  04DE    602          BEQL    30$                     ; Yes if EQL
                       51   5B    D0  04E0    603          MOVL    R11,R1                  ; Get highest locator-2
                          0A    12  04E3    604          BNEQ    20$                     ; If non-zero compare with this
                    51    0A A5    3C  04E5    605          MOVZWL  ED$W_LOC2(R5),R1        ; Get current locator-2
                          04    12  04E9    606          BNEQ    20$                     ; If zero use
                    51    08 A5    3C  04EB    607          MOVZWL  ED$W_LOC1(R5),R1        ; locator-1
                                04EF    608 20$:
                08 A3    51    B1  04EF    609          CMPW    R1,ED$W_LOC1(R3)        ; Does this edit overlap with next?
                          0F    19  04F3    610          BLSS    30$                     ; No if LSS
                       55   53    D0  04F5    611          MOVL    R3,R5                   ; Set next node to be current node
                0A A5    5B    B1  04F8    612          CMPW    R11,ED$W_LOC2(R5)       ; Is locator-2 higher then current
                          CA    18  04FC    613          BGEQ    10$                     ; No if GEQ
                    5B    0A A5    3C  04FE    614          MOVZWL  ED$W_LOC2(R5),R11       ; Reset with higher number
                          C4    11  0502    615          BRB     10$
                                0504    616 30$:
                          53 8ED0   0504    617          POPL    R3                      ; Restore R3
                                05  0507    618          RSB
```

```
                                0508    620          .SBTTL  ACCESS_EDIT
                                0508    621 ;
                                0508    622 ; Subroutine to set up RMS 32 to read an edit from an input file
                                0508    623 ;
                                0508    624 ; Inputs:
                                0508    625 ;       R3 = Edit node address
                                0508    626 ;
                                0508    627 ; Outputs:
                                0508    628 ;       RAB block set up for RFA access to edit lines
                                0508    629 ;
                                0508    630 ;
                                0508    631 ACCESS_EDIT:
                          52    DD  0508    632          PUSHL   R2
 52    0000'CF                  DE  050A    633          MOVAL   W^RANDOM_RAB,R2         ; Set pointer to RAB
           FD9B                  30  050F    634          BSBW    RANDOM_OPEN             ; Get correct file open
         0E 50                   E9  0512    635          BLBC    R0,10$                  ; Error if LBC
10 A2    0E A3                  D0  0515    636          MOVL    ED$W_RFA+0(R3),RAB$W_RFA+0(R2) ; Copy RFA (3 words)
14 A2    12 A3                  B0  051A    637          MOVW    ED$W_RFA+4(R3),RAB$W_RFA+4(R2)
   1E A2    02                  90  051F    638          MOVB    #RAB$C_RFA,RAB$B_RAC(R2)        ; Set into RFA access mode
                                0523    639 10$:
                52 8ED0         0523    640          POPL    R2
                      05        0526    641          RSB
```

```
                     0527   643          .SBTTL  READ_RECORD
                     0527   644 ;
                     0527   645 ; Subroutine to read edit record.  The first record is read from
                     0527   646 ; a random position, subsequent records are read sequentially from
                     0527   647 ; the first.  To achieve this this routine changes the record access
                     0527   648 ; mode from RFA to SEQ after reading the first record
                     0527   649 ;
                     0527   650 ;
                     0527   651 ; Inputs:
                     0527   652 ;       R8 = address of file spec string
                     0527   653 ;       R9 = size of file spec string
                     0527   654 ;
                     0527   655 ; Outputs:
                     0527   656 ;       R6 = Record address
                     0527   657 ;       R7 = Record size
                     0527   658 ;
                     0527   659 ;
                     0527   660 READ_RECORD:
          52    DD   0527   661          PUSHL   R2
52   0000'CF    DE   0529   662          MOVAL   W^RANDOM_RAB,R2         ; Set pointer to RAB
                     052E   663          $GET    RAB = R2                ; Get line
        18 50    E8   0537   664          BLBS    R0,10$                  ; OK if LBS
                     053A   665          ERRMSG  READER,<R8,R9>
           12    11   0550   666          BRB     20$
                     0552   667 10$:
     56    28 A2    D0   0552   668          MOVL    RAB$L_RBF(R2),R6        ; Set buffer address
     57    22 A2    3C   0556   669          MOVZWL  RAB$W_RSZ(R2),R7        ; and line size
     02    1E A2    91   055A   670          CMPB    RAB$B_RAC(R2),#RAB$C_RFA ; Is it in RFA mode?
           04    12   055E   671          BNEQ    20$                     ; No if NEQ
   1E A2    00    90   0560   672          MOVB    #RAB$C_SEQ,RAB$B_RAC(R2) ; Set into SEQ access mode
                     0564   673 20$:
        52 8ED0   0564   674          POPL    R2
           05    0567   675          RSB
```

```
                                0568   677          .SBTTL  RESET_AUDIT
                                0568   678 ;
                                0568   679 ; Subroutine to set new current audit trail string
                                0568   680 ;
                                0568   681 ; Inputs:
                                0568   682 ;       R10 = File node address
                                0568   683 ;
                                0568   684 ; Outputs:
                                0568   685 ;       None
                                0568   686 ;
                                0568   687          .ENABL  LSB
                                0568   688 ;
                                0568   689 ;
                                0568   690 RESET_AUDIT:
                          3C    BB    0568   691          PUSHR   #^M<R2,R3,R4,R5>
        30 AA   18 AA   10 AA   28    056A   692          MOVC3   SLP$Q_AUDDS(R10), -     ; Copy audit string
                                0571   693                  SLP$T_AUDST(R10), -
                                0571   694                  SLP$T_AUCST(R10)
                28 AA   10 AA   B0    0571   695          MOVW    SLP$Q_AUDDS(R10),SLP$Q_AUCDS(R10) ; and string length
                          02    11    0576   696          BRB     10$
                                0578   697 ;
                                0578   698 RESET_CURRENT:
                          3C    BB    0578   699          PUSHR   #^M<R2,R3,R4,R5>
                                057A   700 10$:
  00000000'EF   30 AA   28 AA   28    057A   701          MOVC3   SLP$Q_AUCDS(R10), -     ; Copy audit string
                                0584   702                  SLP$T_AUCST(R10), -
                                0584   703                  SLP_AUDST
              0000'CF   28 AA   B0    0584   704          MOVW    SLP$Q_AUCDS(R10),W^SLP_AUDDS ; and string length
                          3C    BA    058A   705          POPR    #^M<R2,R3,R4,R5>
                          05    058C   706          RSB
                                058D   707 ;
                                058D   708          .DSABL  LSB
```

```
                                058D   710          .SBTTL  WRITE_EDIT
                                058D   711 ;
                                058D   712 ; Subroutine to form and write edit command to output file
                                058D   713 ;
                                058D   714 ; Inputs:
                                058D   715 ;       R2  = Flags
                                058D   716 ;       R3  = Edit node address
                                058D   717 ;       R10 = File node address
                                058D   718 ;
                                058D   719 ; Outputs:
                                058D   720 ;       None
                                058D   721 ;
                                058D   722 ;
                                058D   723 WRITE_EDIT:
      51   0C AA   0000'CF   8D  058D   724          XORB3   W^SLP_FLAGS, -          ; Is audit trail switch for this file
                                0594   725                  SLP$B_FLAGS(R10),R1     ; different to current setting?
            51   0000'8F   B3  0594   726          BITW    #SLPM_AUDIT,R1          ;
                        2C   13  0599   727          BEQL    30$                     ; No if EQL
                     57   01   D0  059B   728          MOVL    #1,R7                   ; Set output line size
     0000'CF   0000'8F   B3  059E   729          BITW    #SLPM_AUDIT,W^SLP_FLAGS ; Was audit ON?
                        0E   12  05A5   730          BNEQ    10$                     ; Yes if NEQ
            56   0000'CF   DE  05A7   731          MOVAL   W^AUDIT_ON,R6           ; It was OFF so turn ON
     0000'CF   0000'8F   A8  05AC   732          BISW    #SLPM_AUDIT,W^SLP_FLAGS
                        12   11  05B3   733          BRB     30$
                                05B5   734 10$:
            56   0000'CF   DE  05B5   735          MOVAL   W^AUDIT_OFF,R6          ; It was ON so turn OFF
     0000'CF   0000'8F   AA  05BA   736          BICW    #SLPM_AUDIT,W^SLP_FLAGS
                                05C1   737 20$:
                  0117   30  05C1   738          BSBW    WRITE_LINE              ; Write command to file
                EE 50    E9  05C4   739          BLBC    R0,10$                  ; Error of LBC
                                05C7   740 30$:
                   57    D4  05C7   741          CLRL    R7                      ; Initialise line size
                   54    D4  05C9   742          CLRL    R4                      ; Zero comment count
0000'CF    0200 8F    3C  05CB   743          MOVZWL  #BUF_SIZE,W^OUTDES      ; Initialise output buffer
0004'CF    0000'CF    DE  05D2   744          MOVAL   W^OUTPUT_BUF,W^OUTDES+4 ; descriptor
        0D 52    02   E1  05D9   745          BBC     #SUM_V_SUPPRESS,R2,32$  ; '--' form of edit?
      0004'DF    2D   90  05DD   746          MOVB    #^A/=/,@W^OUTDES+4      ; Insert first '-'
      0000'CF    01   D0  05E2   747          MOVL    #1,W^OUTLEN
                00DD   30  05E7   748          BSBW    ADJUST_COUNTS
                                05EA   749 32$:
                08 AA   B5  05EA   750          TSTW    SLP$W_LOC1(R10)         ; Is there a locator-1
                   18   13  05ED   751          BEQL    33$                     ; No if EQL
                                05EF   752          $FAO_S  W^LOC_ONE,W^OUTLEN, -   ; Convert and store locator-1
                                05EF   753                  W^OUTDES,SLP$W_LOC1(R10)
                   0A   11  0605   754          BRB     37$
                                0607   755 33$:
      0004'DF    2D   90  0607   756          MOVB    #^A/-/,@W^OUTDES+4      ; Insert a '-'
      0000'CF    01   D0  060C   757          MOVL    #1,W^OUTLEN
                                0611   758 37$:
                00B3   30  0611   759          BSBW    ADJUST_COUNTS
                                0614   760 ;
                0A AA   B5  0614   761          TSTW    SLP$W_LOC2(R10)         ; Is there a locator-2?
                   1B   13  0617   762          BEQL    40$                     ; No if EQL
                                0619   763          $FAO_S  W^LOC_TWO,W^OUTLEN, -   ; Convert and store locator-2
                                0619   764                  W^OUTDES,SLP$W_LOC2(R10)
                0095   30  062F   765          BSBW    ADJUST_COUNTS
                   54   D6  0632   766          INCL    R4                      ; One comma now in line
```

```
                                0634   767 40$:
                     10 AA   B5 0634   768         TSTW    SLP$Q_AUDDS(R10)        ; Audit trail in this command?
                        05   13 0637   769         BEQL    50$                     ; No if EQL
                      FF2C   30 0639   770         BSBW    RESET_AUDIT             ; Reset audit trail
                        15   11 063C   771         BRB     60$
                                063E   772 50$:
                        2D   BB 063E   773         PUSHR   #^M<R0,R2,R3,R5>
28 AA   00  0000'CF  0000'CF  2D 0640   774         CMPC5   W^SLP_AUDDS,W^SLP_AUDST,#0, - ; Is audit trail required for this
                     30 AA      064A
                                064C   775                 SLP$Q_AUCDS(R10),SLP$T_AUCST(R10) ; file equal to current?
                        2D   BA 064C   776         POPR    #^M<R0,R2,R3,R5>
                        35   13 064E   777         BEQL    80$                     ; Yes if EQL
                      FF25   30 0650   778         BSBW    RESET_CURRENT           ; Reset current audit string
                                0653   779 60$:
                  01    54   D1 0653   780         CMPL    R4,#1                   ; One comma on line?
                        10   18 0656   781         BGEQ    70$                     ; Yes if GEQ
              0004'DF   2C   90 0658   782         MOVB    #^A/,/,@W^OUTDES+4      ; Insert comma
              0000'CF   01   D0 065D   783         MOVL    #1,W^OUTLEN
                        63   10 0662   784         BSB     ADJUST_COUNTS
                        54   D6 0664   785         INCL    R4                      ; Increment number of commas
                        EB   11 0666   786         BRB     60$
                                0668   787 70$:
                                0668   788         $FAO_S  W^AUDIT_TRAIL,W^OUTLEN, - ; Insert audit trail
                                0668   789                 W^OUTDES,#SLP_AUDDS
                        44   10 0681   790         BSB     ADJUST_COUNTS
                        54   D6 0683   791         INCL    R4                      ; Increment number of commas on line
                                0685   792 80$:
                     40 AA   B5 0685   793         TSTW    SLP$Q_CMNT(R10)         ; Comment in command?
                        30   13 0688   794         BEQL    110$                    ; No if EQL
                                068A   795 90$:
                  02    54   D1 068A   796         CMPL    R4,#2                   ; Two commas on line
                        10   18 068D   797         BGEQ    100$                    ; Yes if GEQ
              0004'DF   2C   90 068F   798         MOVB    #^A/,/,@W^OUTDES+4      ; Insert comma
              0000'CF   01   D0 0694   799         MOVL    #1,W^OUTLEN
                        2C   10 0699   800         BSB     ADJUST_COUNTS
                        54   D6 069B   801         INCL    R4                      ; Increment number of commas
                        EB   11 069D   802         BRB     90$
                                069F   803 100$:
                51    40 AA   DE 069F   804         MOVAL   SLP$Q_CMNT(R10),R1
                                06A3   805         $FAO_S  W^COMMENTS,W^OUTLEN, -  ; Insert comment into line
                                06A3   806                 W^OUTDES,R1
                        0D   10 06B8   807         BSB     ADJUST_COUNTS
                                06BA   808 110$:
                  01    57   D1 06BA   809         CMPL    R7,#1                   ; Is line only '-'?
                        07   13 06BD   810         BEQL    120$                    ; Yes if EQL so dont output it
              56  0000'CF   DE 06BF   811         MOVAL   W^OUTPUT_BUF,R6         ; Set buffer pointer
                        15   10 06C4   812         BSB     WRITE_LINE
                                06C6   813 120$:
                        05   05 06C6   814         RSB
                                06C7   815 ;
                                06C7   816 ;
                                06C7   817 ;
                                06C7   818 ; Subroutine to adjust output buffer descriptor as each
                                06C7   819 ; element of command line is inserted
                                06C7   820 ;
                                06C7   821 ;
                                06C7   822 ; Inputs:
```

```
                     06C7    823 ;       R7 = Buffer size
                     06C7    824 ;       OUTLEN = Length of last entry
                     06C7    825 ;
                     06C7    826 ; Outputs:
                     06C7    827 ;       R7 = New buffer length
                     06C7    828 ;       OUTDES = Describes remainder of buffer
                     06C7    829 ;
                     06C7    830 ;
                     06C7    831 ADJUST_COUNTS:
     57  0000'CF  A0 06C7    832         ADDW2   W^OUTLEN,R7             ; Form new line length
0000'CF  0000'CF  A2 06CC    833         SUBW2   W^OUTLEN,W^OUTDES       ; Reduce size of remaining buffer
0004'CF  0000'CF  C0 06D3    834         ADDL2   W^OUTLEN,W^OUTDES+4     ; Set pointer to first unused byte
                  05 06DA    835         RSB
```

```
                                06DB   837          .SBTTL  WRITE_LINE
                                06DB   838 ;
                                06DB   839 ; Subroutine to write line to SLPPR output file
                                06DB   840 ;
                                06DB   841 ; Inputs:
                                06DB   842 ;       R6 = Line buffer address
                                06DB   843 ;       R7 = Line size
                                06DB   844 ;
                                06DB   845 ;
                                06DB   846 ; Outputs:
                                06DB   847 ;       R0 = success/error status code
                                06DB   848 ;
                                06DB   849 ;
                                06DB   850 WRITE_LINE:
                             52 DD  06DB   851          PUSHL   R2
0000'CF    0000'8F B3  06DD   852          BITW    #MERM_OUTPUT,W^MERGE_FLAGS ; Was output file specified?
                             37 13  06E4   853          BEQL    10$                         ; No if EQL
                52   0000'CF DE  06E6   854          MOVAL   W^OUTPUT_RAB,R2             ; Get address of RAB
                  28 A2  56 D0  06EB   855          MOVL    R6,RAB$L_RBF(R2)            ; Insert line address
                  22 A2  57 B0  06EF   856          MOVW    R7,RAB$W_RSZ(R2)            ; and size
                         06F3   857          $PUT    RAB = R2
                        1E 50  E8  06FC   858          BLBS    R0,10$                      ; OK if LBS
                        0C A2  DD  06FF   859          PUSHL   RAB$L_STV(R2)               ; Signal error
                           50  DD  0702   860          PUSHL   R0
                               0704   861          ERRMSG  OUTERR
00000000'GF                02  FB  0716   862          CALLS   #2,G^LIB$SIGNAL
                               C71D   863 10$:
                           52 8ED0 0720   864          POPL    R2
                              05   0723   865          RSB
```

```
                                0721   867          .SBTTL  COMMAND_CHECK
                                0721   868 ;
                                0721   869 ; Subroutine to check if line is a command
                                0721   870 ;
                                0721   871 ; Inputs:
                                0721   872 ;       R6 = Address of line
                                0721   873 ;       R7 = Size of line
                                0721   874 ;       R10= File address
                                0721   875 ;
                                0721   876 ; Outputs:
                                0721   877 ;       R4[00] = 0:Data  1:Command
                                0721   878 ;       R4[01] = 0:Normal command  1:Data terminator command
                                0721   879 ;       R4[02] = 0:Data terminator  1:End of edit
                                0721   880 ;       R6 = Address of line
                                0721   881 ;       R7 = Size of line
                                0721   882 ;
                                0721   883          $TPADEF
                                0721   884 ;
                                0721   885 COMMAND_CHECK:
0000'CF    0E AA    B0          0721   886          MOVW    SLP$W_DOT(R10),W^DOT    ; Get dot value for this file
0008'CF    0C AA    9B          0727   887          MOVZBW  SLP$B_FLAGS(R10),W^FLAGS ; Get current flags word
0008'CF    0C 8F    8A          072D   888          BICB2   #SLPM_SUPPRESS,W^FLAGS  ; but clear suppress flag
         0002'CF    B4          0733   889          CLRW    W^LOC
         0004'CF    B4          0737   890          CLRW    W^LOC1
         0006'CF    B4          073B   891          CLRW    W^LOC2
         0016'CF    7C          073F   892          CLRQ    W^AUDIT
         001E'CF    7C          0743   893          CLRQ    W^COMMENT
    000E'CF    56    D0          0747   894          MOVL    R6,W^ADDR               ; Save line address and size
    0012'CF    57    D0          074C   895          MOVL    R7,W^SIZE
                                0751   896 ;
    0032'CF    56    D0          0751   897          MOVL    R6,W^TPARSE_BLOCK+TPA$L_STRINGPTR ; Set TPARSE input
    002E'CF    57    D0          0756   898          MOVL    R7,W^TPARSE_BLOCK+TPA$L_STRINGCNT       ; string descriptor
         0000'CF    DF          075B   899          PUSHAL  W^SUM_MER_KEY
         0000'CF    DF          075F   900          PUSHAL  W^SUM_MER_STATE
         0026'CF    DF          0763   901          PUSHAL  W^TPARSE_BLOCK
00000000'GF    03    FB          0767   902          CALLS   #3,G^LIB$TPARSE
              47 50    E9        076E   903          BLBC    R0,20$                  ; Error if LBC
    56    000E'CF    D0          0771   904          MOVL    W^ADDR,R6               ; Reset line address and size
    57    0012'CF    D0          0776   905          MOVL    W^SIZE,R7
 0E AA    0000'CF    B0          077B   906          MOVW    W^DOT,SLP$W_DOT(R10)
 0C AA    0008'CF    90          0781   907          MOVB    W^FLAGS,SLP$B_FLAGS(R10)
 08 AA    0004'CF    B0          0787   908          MOVW    W^LOC1,SLP$W_LOC1(R10)
 0A AA    0006'CF    B0          078D   909          MOVW    W^LOC2,SLP$W_LOC2(R10)
0010 CA   0016'CF    B0          0793   910          MOVW    W^AUDIT,W^SLP$Q_AUDDS(R10)
          001A'CF    D5          079A   911          TSTL    W^AUDIT+4               ; Was audit trail specified?
                0D   13          079E   912          BEQL    10$                     ; No if EQL
                3F   BB          07A0   913          PUSHR   #^M<R0,R1,R2,R3,R4,R5>
14 BA     001A'DF    0016'CF     28     07A2   914          MOVC3   W^AUDIT,@W^AUDIT+4,@SLP$Q_AUDDS+4(R10)
                3F   BA          07AB   915          POPR    #^M<R0,R1,R2,R3,R4,R5>
                                07AD   916 10$:
 40 AA    001E'CF    7D          07AD   917          MOVQ    W^COMMENT,SLP$Q_CMNT(R10)
    54    000A'CF    D0          07B3   918          MOVL    W^TYPE,R4
                                07B8   919 20$:
                05   05          07B8   920          RSB
                                07B9   921
                                00000000   922          .PSECT  $DATA,NOEXE,WRT
                                0000   923 ;
```

```
                        0000    924 ; Local data
                        0000    925 ;
                0000    0000    926 DOT:        .WORD   0                       ; '.' value
                0000    0002    927 LOC:        .WORD   0
                0000    0004    928 LOC1:       .WORD   0
                0000    0006    929 LOC2:       .WORD   0
                0000    0008    930 FLAGS:      .WORD   0
            00000000    000A    931 TYPE:       .LONG   0
            00000000    000E    932 ADDR:       .LONG   0
            00000000    0012    933 SIZE:       .LONG   0
    00000000 00000000   0016    934 AUDIT:      .QUAD   0
    00000000 00000000   001E    935 COMMENT:    .QUAD   0
```

```
                     0026   937          .SBTTL  TPARSE
                     0026   938 ;
                     0026   939 ;
                     0026   940 TPARSE_BLOCK:
00000008             0026   941          .LONG   TPA$K_COUNT0
000000AA             002A   942          .BLKL   TPA$K_LENGTH0-4
                     00AA   943 ;
0000002C             00AA   944          COMMA = ^X2C
0000003B             00AA   945          SEMICOLON = ^X3B
                     00AA   946 ;
                     00AA   947          .SAVE
                     00AA   948 ;
                     00AA   949          $INIT_STATE     SUM_MER_STATE,SUM_MER_KEY
                     00AA   950 ;
                     00AA   951 ; Get 1st character of line
                     00AA   952 ;
                     00AA   953          $STATE
                     00AA   954          $TRAN   TPA$_LAMBDA,,ACT_BLANKS_SIG
                     00AA   955          $STATE
                     00AA   956          $TRAN   '%',CMND,ACT_PERCENT
                     00AA   957          $TRAN   '-',EDIT
                     00AA   958          $TRAN   '/',TERM
                     00AA   959          $TRAN   '@',TPA$_FAIL,ACT_ATSIGN,,,0
                     00AA   960          $TRAN   '\',CMND,ACT_BACKSLASH
                     00AA   961          $TRAN   TPA$_EOS,DATA
                     00AA   962          $TRAN   TPA$_ANY,DATA
                     00AA   963 ;
                     00AA   964 ; End data line
                     00AA   965 ;
                     00AA   966          $STATE  DATA
                     00AA   967          $TRAN   TPA$_LAMBDA,TPA$_EXIT,ACT_EXIT,,,0
                     00AA   968 ;
                     00AA   969 ; End normal command line
                     00AA   970 ;
                     00AA   971          $STATE  CMND
                     00AA   972          $TRAN   TPA$_LAMBDA,TPA$_EXIT,ACT_EXIT,,,1
                     00AA   973 ;
                     00AA   974 ; End data terminating command
                     00AA   975 ;
                     00AA   976          $STATE  TERM
                     00AA   977          $TRAN   TPA$_LAMBDA,TPA$_EXIT,ACT_EXIT,,,7
                     00AA   978 ;
                     00AA   979 ;
                     00AA   980 ; Edit command
                     00AA   981 ;
                     00AA   982 ; Read locator-1
                     00AA   983 ;
                     00AA   984          $STATE  EDIT
                     00AA   985          $TRAN   '-',,ACT_SUPPRESS
                     00AA   986          $TRAN   TPA$_LAMBDA
                     00AA   987          $STATE
                     00AA   988          $TRAN   TPA$_LAMBDA,,ACT_BLANKS_NSIG
                     00AA   989          $STATE
                     00AA   990          $TRAN   !LOCATOR,,ACT_LOC1
                     00AA   991 ;
                     00AA   992 ; Read Locator-2
                     00AA   993 ;
```

```
                        00AA   994          $STATE
                        00AA   995          $TRAN   TPA$_EOS,TPA$_EXIT
                        00AA   996          $TRAN   SEMICOLON,CMNT,ACT_CMNT
                        00AA   997          $TRAN   COMMA
                        00AA   998          $STATE
                        00AA   999          $TRAN   !LOCATOR,,ACT_LOC2
                        00AA  1000          $TRAN   TPA$_EOS,TPA$_EXIT
                        00AA  1001 ;
                        00AA  1002 ; Read audit string
                        00AA  1003 ;
                        00AA  1004          $STATE
                        00AA  1005          $TRAN   TPA$_EOS,TPA$_EXIT
                        00AA  1006          $TRAN   SEMICOLON,CMNT,ACT_CMNT
                        00AA  1007          $TRAN   COMMA
                        00AA  1008          $STATE
                        00AA  1009          $TRAN   '/',,ACT_AUDIT
                        00AA  1010          $TRAN   TPA$_EOS,TPA$_EXIT
                        00AA  1011          $TRAN   SEMICOLON,CMNT,ACT_CMNT
                        00AA  1012          $STATE  AUDCH
                        00AA  1013          $TRAN   '/',,ACT_AUDEND
                        00AA  1014          $TRAN   TPA$_ANY,AUDCH,ACT_AUDCH
                        00AA  1015 ;
                        00AA  1016 ; Read comment line
                        00AA  1017 ;
                        00AA  1018          $STATE
                        00AA  1019          $TRAN   TPA$_EOS,TPA$_EXIT
                        00AA  1020          $TRAN   SEMICOLON,CMNT,ACT_CMNT
                        00AA  1021          $STATE  CMNT
                        00AA  1022          $TRAN   TPA$_LAMBDA,TPA$_EXIT
                        00AA  1023 ;
                        00AA  1024 ;
                        00AA  1025 ; Subexpression to parse locator
                        00AA  1026 ;
                        00AA  1027          $STATE  LOCATOR
                        00AA  1028          $TRAN   '.',,ACT_DOT
                        00AA  1029          $TRAN   TPA$_DECIMAL,,ACT_LOCNUM
                        00AA  1030          $TRAN   TPA$_LAMBDA,TPA$_EXIT
                        00AA  1031          $STATE
                        00AA  1032          $TRAN   '+'
                        00AA  1033          $TRAN   TPA$_LAMBDA,TPA$_EXIT
                        00AA  1034          $STATE
                        00AA  1035          $TRAN   TPA$_DECIMAL,,ACT_PLUS
                        00AA  1036          $STATE
                        00AA  1037          $TRAN   TPA$_LAMBDA,TPA$_EXIT
                        00AA  1038 ;
                        00AA  1039          $END_STATE
                        00AA  1040 ;
                        00AA  1041 ;
                   000000AA  1042          .RESTORE
```

```
                                00AA  1044 ;
                                00AA  1045 ; Tparse action routines
                                00AA  1046 ;
                                00AA  1047 ;
                           000007B9  1048         .PSECT  $CODE,NOWRT,EXE
                                07B9  1049
                                07B9  1050 ACT_BLANKS_SIG:
                              0000    07B9  1051         .WORD   0
                 00 04 AC     00  E2  07BB  1052         BBSS    #TPA$V_BLANKS,TPA$L_OPTIONS(AP),10$
                                07C0  1053 10$:
                                04    07C0  1054         RET
                                07C1  1055 ;
                                07C1  1056 ;
                                07C1  1057 ACT_BLANKS_NSIG:
                              0000    07C1  1058         .WORD   0
                 00 04 AC     00  E5  07C3  1059         BBCC    #TPA$V_BLANKS,TPA$L_OPTIONS(AP),10$
                                07C8  1060 10$:
                                04    07C8  1061         RET
                                07C9  1062 ;
                                07C9  1063 ;
                                07C9  1064 ACT_PERCENT:
                              0000    07C9  1065         .WORD   0
        0008'CF    0000'8F      A8  07CB  1066         BISW    #SLPM_AUDIT,W^FLAGS     ; Switch on audit trail
                                04    07D2  1067         RET
                                07D3  1068 ;
                                07D3  1069 ;
                                07D3  1070 ACT_ATSIGN:
                              0000    07D3  1071         .WORD   0
                                07D5  1072         ERRMSG  ATSIGN,<ADDR,SIZE>      ; Print error line
          000A'CF    20 AC      DE  07F3  1073         MOVAL   TPA$L_PARAM(AP),W^TYPE
                50   01        D0  07F9  1074         MOVL    #1,R0
                                04    07FC  1075         RET
                                07FD  1076 ;
                                07FD  1077 ;
                                07FD  1078 ACT_BACKSLASH:
                              0000    07FD  1079         .WORD   0
        0008'CF    0000'8F      AA  07FF  1080         BICW    #SLPM_AUDIT,W^FLAGS     ; Switch off audit trail
                                04    0806  1081         RET
                                0807  1082 ;
                                0807  1083 ;
                                0807  1084 ACT_EXIT:
                              0000    0807  1085         .WORD   0
          000A'CF    20 AC      D0  0809  1086         MOVL    TPA$L_PARAM(AP),W^TYPE          ; Set return type
                                04    080F  1087         RET
                                0810  1088 ;
                                0810  1089 ;
                                0810  1090 ACT_LOC1:
                              0000    0810  1091         .WORD   0
             000A'CF    01      B0  0812  1092         MOVW    #1,W^TYPE               ; Assume normal command
        0004'CF    0002'CF      B0  0817  1093         MOVW    W^LOC,W^LOC1
                        05     13  081E  1094         BEQL    10$                     ; If EQL is a normal command
             000A'CF    03      B0  0820  1095         MOVW    #3,W^TYPE               ; Set as data terminator command
                                0825  1096 10$:
                   0002'CF      B4  0825  1097         CLRW    W^LOC
                                04    0829  1098         RET
                                082A  1099 ;
                                082A  1100 ;
```

```
                            082A  1101 ACT_LOC2:
                      0000  082A  1102         .WORD   0
          0006'CF   0002'CF   B0  082C  1103         MOVW    W^LOC,W^LOC2
                        04    0833  1104         RET
                              0834  1105 ;
                              0834  1106 ;
                              0834  1107 ACT_DOT:
                      0000  0834  1108         .WORD   0
          0002'CF   0000'CF   B0  0836  1109         MOVW    W^DOT,W^LOC
                        04    083D  1110         RET
                              083E  1111 ;
                              083E  1112 ;
                              083E  1113 ACT_LOCNUM
                      0000  083E  1114         .WORD   0
            0002'CF   1C AC   B0  0840  1115         MOVW    TPA$L_NUMBER(AP),W^LOC
          0000'CF   0002'CF   B0  0846  1116         MOVW    W^LOC,W^DOT
                        04    084D  1117         RET
                              084E  1118 ;
                              084E  1119 ;
                              084E  1120 ACT_PLUS:
                      0000  084E  1121         .WORD   0
            0002'CF   1C AC   A0  0850  1122         ADDW2   TPA$L_NUMBER(AP),W^LOC
          0000'CF   0002'CF   B0  0856  1123         MOVW    W^LOC,W^DOT
                        04    085D  1124         RET
                              085E  1125
                              085E  1126 ;
                              085E  1127 ;
                              085E  1128 ACT_AUDIT:
                      0000  085E  1129         .WORD   0
          001A'CF   0C AC   D0  0860  1130         MOVL    TPA$L_STRINGPTR(AP),W^AUDIT+4
            00 04 AC    00   E2  0866  1131         BBSS    #TPA$V_BLANKS,TPA$L_OPTIONS(AP),10$ ; Make blanks significant
                              086B  1132 10$:
                        04    086B  1133         RET
                              086C  1134 ;
                              086C  1135 ;
                              086C  1136 ACT_AUDCH:
                      0000  086C  1137         .WORD   0
                0016'CF   D6  086E  1138         INCL    W^AUDIT
                        04    0872  1139         RET
                              0873  1140 ;
                              0873  1141 ;
                              0873  1142 ACT_AUDEND:
                      0000  0873  1143         .WORD   0
            00 04 AC    00   E5  0875  1144         BBCC    #TPA$V_BLANKS,TPA$L_OPTIONS(AP),10$ ; Switch off blank processing
                              087A  1145 10$:
                        04    087A  1146         RET
                              087B  1147 ;
                              087B  1148 ;
                              087B  1149 ACT_CMNT:
                      0000  087B  1150         .WORD   0
          001E'CF   08 AC   7D  087D  1151         MOVQ    TPA$L_STRINGCNT(AP),W^COMMENT
                        04    0883  1152         RET
                              0884  1153 ;
                              0884  1154 ;
                              0884  1155 ACT_SUPPRESS::
                      0000  0884  1156         .WORD   0
          0008'CF   00'8F   88  0886  1157         BISB2   #SLPM_SUPPRESS,W^FLAGS  ; set clash messages suppressed flag
```

```
                04  088C  1158          RET
```

```
088D  1160 ;
088D  1161 ;
088D  1162          .END
```

```
$$                      = 00000002
$$$CNT                  = 00000003
$$$FLG                  = FFFFFFFF
$$$KEY                  = FFFFFFFF
$$$KFG                  = FFFFFFFF
$$$MOD                  = 00000000
$$.TMP1                 = 00000001
$$.TMP2                 = 00000052
$$KEYTAB                = 00000000 R      05
$$T2                    = 00000004
..AFLG                  = 00000000
..FLG                   = 00000002
..TYP                   = 00000054
.LEN                    = 00000004
ACCESS_EDIT               00000508 R      02
ACT_ATSIGN                000007D3 R      02
ACT_AUDCH                 0000086C R      02
ACT_AUDEND                00000873 R      02
ACT_AUDIT                 0000085E R      02
ACT_BACKSLASH             000007FD R      02
ACT_BLANKS_NSIG           000007C1 R      02
ACT_BLANKS_SIG            000007B9 R      02
ACT_CMNT                  0000087B R      02
ACT_DOT                   00000834 R      02
ACT_EXIT                  00000807 R      02
ACT_LOC1                  00000810 R      02
ACT_LOC2                  0000082A R      02
ACT_LOCNUM                0000083E R      02
ACT_PERCENT               00000?C9 R      02
ACT_PLUS                  0000084E R      02
ACT_SUPPRESS              00000884 RG     02
ADDR                      0000000E R      03
ADJUST_COUNTS             000006C7 R      02
AUDCH                     000000AA R      04
AUDIT                     00000016 R      03
AUDIT_OFF                 ********   X    02
AUDIT_ON                  ********   X    02
AUDIT_TRAIL               ********   X    02
BIT...                  = 00000004
BUF_SIZE                = 00000200
CLASH_LINE                0000023C R      02
CMD_SIZE                = 00000084
CMND                      00000040 R      04
CMNT                      000000C4 R      04
COMMA                   = 0000002C
COMMAND_CHECK             00000721 R      02
COMMENT                   0000001E R      03
COMMENTS                  ********   X    02
DATA                      00000033 R      04
DOT                       00000000 R      03
ED$B_FILENO             = 00000019
ED$B_FLAGS              = 00000018
ED$K_BLN                = 0000001A
ED$L_BWD                = 00000004
ED$L_FILE               = 00000014
ED$L_FWD                = 00000000
ED$W_LINES              = 0000000C
ED$W_LOC1               = 00000008
ED$W_LOC2               = 0000000A
ED$W_RFA                = 0000000E
EDIT                      0000005A R      04
EDIT_NODES                ********   X    02
ED_SIZE                   ********   X    02
END_EDIT                  ********   X    02
FAB$L_NAM               = 00000C28
FAB$L_STV               = 0000000C
FILE_NODES                ********   X    02
FLAGS                     00000008 R      03
GET_NODE                  0000017B R      02
INPUT_FAB                 ********   X    02
INPUT_RAB                 ********   X    02
INSERT_NODE               000001A8 R      02
LIB$FREE_VM               ********   X    02
LIB$GET_VM                ********   X    02
LIB$SIGNAL                ********   X    02
LIB$TPARSE                ********   X    02
LOC                       00000002 R      03
LOC1                      00000004 R      03
LOC2                      00000006 R      03
LOCATOR                   000000C8 R      04
LOC_ONE                   ********   X    02
LOC_TWO                   ********   X    02
MER$_ATSIGN               ********   X    02
MER$_CLASH                ********   X    02
MER$_CLOSER               ********   X    02
MER$_CLSHLN               ********   X    02
MER$_CONNEC               ********   X    02
MER$_DISCON               ********   X    02
MER$_INPEOF               ********   X    02
MER$_OUTERR               ********   X    02
MER$_READER               ********   X    02
MER$_REOPEN               ********   X    02
MER$_SYNTAX               ********   X    02
MERGE_FILES               00000000 RG     02
MERGE_FLAGS               ********   X    02
MERM_OUTPUT               ********   X    02
NAM$B_RSL               = 00000003
NAM$K_BLN               = 00000060
NAM$L_RSA               = 00000004
OUTDES                    ********   X    02
OUTLEN                    ********   X    02
OUTPUT_BUF                ********   X    02
OUTPUT_RAB                ********   X    02
PASS_ONE                  0000000B R      02
PASS_TWO                  000003F9 R      02
PROCESS_EDIT              00000421 R      02
RAB$B_RAC               = 0000001E
RAB$C_RFA               = 00000002
RAB$C_SEQ               = 00000000
RAB$L_RBF               = 00000028
RAB$L_STV               = 0000000C
RAB$W_RFA               = 00000010
RAB$W_RSZ               = 00000022
RANDOM_FAB                ********   X    02
```

```
RANDOM_FILE                     ********    X   02
RANDOM_OPEN                     000002AD R      02
RANDOM_RAB                      ********    X   02
RANGE_EDIT                      000004BC R      02
READ_LINE                       000003A4 R      02
READ_RECORD                     00000527 R      02
REPORT_CLASH                    000001F6 R      02
RESET_AUDIT                     00000568 R      02
RESET_CURRENT                   00000578 R      02
RETURN_NODE                     00000193 R      02
RMS$_EOF                        ********    X   02
SEMICOLON                     = 0000003B
SET_UP_NODES                    000000EF R      02
SIZ...                        = 00000001
SIZE                            00000012 R      03
SLPSB_FILENO                  = 0000000D
SLPSB_FLAGS                   = 0000000C
SLPSK_BLN                     = 000000A8
SLPSL_BWD                     = 00000004
SLPSL_FWD                     = 00000000
SLPSQ_AUCDS                   = 00000028
SLPSQ_AUDDS                   = 00000010
SLPSQ_CMNT                    = 00000040
SLPST_AUCST                   = 00000030
SLPST_AUDST                   = 00000018
SLPST_NAM                     = 00000048
SLPSW_DOT                     = 0000000E
SLPSW_LOC1                    = 00000008
SLPSW_LOC2                    = 0000000A
SLPM_AUDIT                      ********    X   02
SLPM_SUPPRESS                   ********    X   02
SLPV_SUPPRESS                   ********    X   02
SLP_AUDDS                       ********    X   02
SLP_AUDST                       ********    X   02
SLP_FLAGS                       ********    X   02
SUM_MER_KEY                     00000000 RG     05
SUM_MER_STATE                   00000000 RG     04
SUM_M_1STEDIT                 = 00000002
SUM_M_EXPED                   = 00000001
SUM_M_SPRCDE                  = 00000008
SUM_M_SUPPRESS                = 00000004
SUM_V_1STEDIT                 = 00000001
SUM_V_EXPED                   = 00000000
SUM_V_SPRCDE                  = 00000003
SUM_V_SUPPRESS                = 00000002
SYS$CLOSE                       ********   GX   02
SYS$CONNECT                     ********   GX   02
SYS$DISCONNECT                  ********   GX   02
SYS$FAO                         ********    X   02
SYS$GET                         ********   GX   02
SYS$OPEN                        ********   GX   02
SYS$PUT                         ********   GX   02
TERM                            0000004D R      04
TPASK_COUNTO                  = 00000008
TPASK_LENGTHO                 = 00000024
TPASL_NUMBER                  = 0000001C
TPASL_OPTIONS                 = 00000004
TPASL_PARAM                   = 00000020
TPASL_STRINGCNT               = 00000008
TPASL_STRINGPTR               = 0000000C
TPASV_BLANKS                  = 00000000
TPAS_ALPHA                    = 000001EE
TPAS_ANY                      = 000001ED
TPAS_BLANK                    = 000001F2
TPAS_DECIMAL                  = 000001F3
TPAS_DIGIT                    = 000001EF
TPAS_EOS                      = 000001F7
TPAS_EXIT                     = FFFFFFFF
TPAS_FAIL                     = FFFFFFFE
TPAS_FILESPEC                 = 000001EA
TPAS_HEX                      = 000001F5
TPAS_IDENT                    = 000001EC
TPAS_KEYWORD                  = 00000100
TPAS_LAMBDA                   = 000001F6
TPAS_MAXKEY                   = 000000DC
TPAS_OCTAL                    = 000001F4
TPAS_STRING                   = 000001F0
TPAS_SUBXPR                   = 000001F8
TPAS_SYMBOL                   = 000001F1
TPAS_UIC                      = 000001EB
TPARSE_BLOCK                    00000026 R      03
TYPE                            0000000A R      03
VIRT_ADDR                       ********    X   02
WRITE_EDIT                      0000058D R      02
WRITE_LINE                      000006DB R      02
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
$CODE	0000088D (2189.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE
$DATA	000000AA (170.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC BYTE
_LIB$STATE$	000000E8 (232.)	04 (4.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC BYTE
_LIB$KEY0$	00000000 (0.)	05 (5.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC WORD
_LIB$KEY1$	00000000 (0.)	06 (6.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC WORD

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	37	00:00:00.08	00:00:00.36
Command processing	132	00:00:00.71	00:00:04.69
Pass 1	441	00:00:21.50	00:00:49.99
Symbol table sort	5	00:00:01.22	00:00:03.38
Pass 2	212	00:00:05.22	00:00:11.81
Symbol table output	24	00:00:00.18	00:00:00.20
Psect synopsis output	3	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	856	00:00:28.95	00:01:10.46

The working set limit was 1800 pages.
114014 bytes (223 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 847 non-local and 86 local symbols.
1247 source lines were read in Pass 1, producing 33 object records in Pass 2.
54 pages of virtual memory were used to define 40 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	0
_255DUA28:[SYSLIB]STARLET.MLB;2	28
TOTALS (all libraries)	28

1243 GETS were required to define 28 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:SUMED/OBJ=OBJ$:SUMED MSRC$:SUMCOM/UPDATE=(ENH$:SUMCOM)+MSRC$:SUMED/UPDATE=(ENH$:SUMED)+EXECML$/LIB
```

SUMSLP
MAP

SRTTRNMSG
LIS

DATA
LIS

SUMSHR
MAP

SUMDATA
LIS

UTLCHKCLI
LIS

SUM

SUM
MAP

SUMDEF
MAR

MAIN
LIS

SUMED
LIS

SUMEDIT
LIS

GETCMD
LIS