```
RRRRRRRRRRR      MMM          MMM      SSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSSSSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRRRRRRRRRR      MMM          MMM      SSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSS
RRR    RRR       MMM          MMM               SSS
RRR    RRR       MMM          MMM               SSS
RRR    RRR       MMM          MMM               SSS
RRR       RRR    MMM          MMM               SSS
RRR       RRR    MMM          MMM               SSS
RRR       RRR    MMM          MMM               SSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR       RRR    MMM          MMM               SSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
```

```
RRRRRRRR    MM      MM    000000    FFFFFFFFFF    IIIIII    LL          FFFFFFFFFF  NN      NN    CCCCCCCC
RRRRRRRR    MM      MM    000000    FFFFFFFFFF    IIIIII    LL          FFFFFFFFFF  NN      NN    CCCCCCCC
RR      RR  MMMM  MMMM  00      00  FF              II      LL          FF          NN      NN  CC
RR      RR  MMMM  MMMM  00      00  FF              II      LL          FF          NN      NN  CC
RR      RR  MM  MM  MM  00    0000  FF              II      LL          FF          NNNN    NN  CC
RR      RR  MM  MM  MM  00    0000  FF              II      LL          FF          NNNN    NN  CC
RRRRRRRR    MM      MM  00  00  00  FFFFFFFF        II      LL          FFFFFFFF    NN  NN  NN  CC
RRRRRRRR    MM      MM  00  00  00  FFFFFFFF        II      LL          FFFFFFFF    NN  NN  NN  CC
RR  RR      MM      MM  0000    00  FF              II      LL          FF          NN    NNNN  CC
RR  RR      MM      MM  0000    00  FF              II      LL          FF          NN    NNNN  CC
RR    RR    MM      MM  00      00  FF              II      LL          FF          NN      NN  CC              ....
RR    RR    MM      MM  00      00  FF              II      LL          FF          NN      NN  CC              ....
RR      RR  MM      MM    000000    FF            IIIIII    LLLLLLLLLL  FF          NN      NN    CCCCCCCC  ....
RR      RR  MM      MM    000000    FF            IIIIII    LLLLLLLLLL  FF          NN      NN    CCCCCCCC  ....


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
```

Table of contents

```
0000     1          $BEGIN  RMOFILFNC,000,RMS$RMSO,<FILE FUNCTIONS MODULE>
0000     2
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 ; Facility: rms32
0000    29 ;
0000    30 ; Abstract:
0000    31 ;       this module includes various routines to
0000    32 ;       perform files-11 primitive functions.
0000    33 ;
0000    34 ;
0000    35 ; Environment:
0000    36 ;               star processor running starlet exec.
0000    37 ;
0000    38 ; Author:       L F Laverdure,   creation date: 11-march-77
0000    39 ;
0000    40 ; Modified By:
0000    41 ;
0000    42 ;       V03-042 RAS0279         Ron Schaefer            26-Mar-1984
0000    43 ;               Add additional system errors to RM$MAPERR:
0000    44 ;                       SS$_IVDEVNAM --> RMS$_DEV
0000    45 ;                       SS$_DEVALLOC --> RMS$_DNR
0000    46 ;
0000    47 ;       V03-041 DGB0019         Donald G. Blair         05-Mar-1984
0000    48 ;               Completely reorganize RM$FILFNC and its alternate
0000    49 ;               entry points as part of the restructuring necessary
0000    50 ;               to implement access mode protected files.
0000    51 ;
0000    52 ;       V03-040 KBT0207         Keith B. Thompson       23-Aug-1982
0000    53 ;               Reorganize psects
0000    54 ;
0000    55 ;       V03-039 KPL0013         Peter Lieberwirth        4-Feb-1982
0000    56 ;               Map DEADLOCK and EXENQLM system error messages.
0000    57 ;
```

```
0000     58 ;       V03-038 KPL0012         Peter Lieberwirth       19-Aug-1981
0000     59 ;               Special case some system error messages from the lock
0000     60 ;               manager in RM$MAPERR.  Also, reorder somewhat the entries
0000     61 ;               in the table, putting more likely errors in front.
0000     62 ;
0000     63 ;       V03-037 CDS0002         C Saether       18-Jun-1981
0000     64 ;               Add RMS error codes definitions.
0000     65 ;
0000     66 ;       V03-036 KPL0011         Peter Lieberwirth       8-Jun-1981
0000     67 ;               Add $ENQ codes to maptable for RM$MAPERR.
0000     68 ;
0000     69 ;       V03-035 CDS0001         C Saether       29-Dec-1980     15:10
0000     70 ;               Remove code for ACT error (not referenced).
0000     71 ;
0000     72 ;       V034    REFORMAT        Ken Henderson   29-JUL-1980     16:18
0000     73 ;               code was reformatted
0000     74 ;
0000     75 ;       V033                    Tim Halvorsen   18-SEP-1979
0000     76 ;               extracted rm$setdid into rm0setdid module.
0000     77 ;
0000     78 ;       V032    RAN0003         L F Laverdure   10-OCT-1978     13:40
0000     79 ;               file sharing code modifications
0000     80 ;
0000     81 ; Revision History:
0000     82 ;
0000     83 ;       L F Laverdure,      10-OCT-1978  13:40  ; file sharing modifications
0000     84 ;
0000     85 ;       L F Laverdure,      11-AUG-1978  17:14  ; add rm$fcpfnc_alt5 entry point
0000     86 ;
0000     87 ;       L F Laverdure,      10-JUL-1978  11:51
0000     88 ;               mod's for mbf support for seq files
0000     89 ;--
0000     90 ;
0000     91
```

```
0000     93          .SBTTL  DECLARATIONS
0000     94
0000     95 ;
0000     96 ; Macros:
0000     97 ;
0000     98
0000     99          $IODEF
0000    100          $SSDEF
0000    101          $FABDEF
0000    102          $FWADEF
0000    103          $IFBDEF
0000    104          $IRBDEF
0000    105          $FIBDEF
0000    106          $QIODEF
0000    107          $RMSDEF
```

```
0000   109          .SBTTL  RM$FCPFNC - ROUTINE TO ISSUE A FILE FUNCTION QIO
0000   110
0000   111 ;++
0000   112 ; RM$FCPFNC
0000   113 ; RM$FCPEXTEND
0000   114 ; RM$FCPFNC_P4
0000   115 ; RM$FCPFNC_NOFIB
0000   116 ;
0000   117 ; this routine completes the qio argument list,
0000   118 ; issues the qio, and awaits completion of
0000   119 ; the function.
0000   120 ;
0000   121 ; Calling sequence:
0000   122 ;
0000   123 ;       bsbw    rm$fcpfnc
0000   124 ;       bsbw    rm$fcpextend
0000   125 ;       bsbw    rm$fcpfnc_p4
0000   126 ;       bsbw    rm$fcpfnc_nofib
0000   127 ;
0000   128 ; Input Parameters:
0000   129 ;
0000   130 ;       r11     impure area address
0000   131 ;       r10     fwa address (input for rm$fcpfnc)
0000   132 ;       r9      ifab/irab address
0000   133 ;       r8      fab/rab address
0000   134 ;       r0      qio function code (not an input for rm$fcpextend)
0000   135 ;
0000   136 ; Implicit Inputs:
0000   137 ;
0000   138 ;       ifb$l_chnl      -  i/o channel
0000   139 ;       fwa$q_fib       -  fib descriptor (and buffer)
0000   140 ;                       -  (note: input for entry at rm$fcpfnc only)
0000   141 ;       see also documentation preceding each entry point
0000   142 ;
0000   143 ; outputs:
0000   144 ;
0000   145 ;       r0      status code
0000   146 ;       r1-r4,ap destroyed
0000   147 ;
0000   148 ; Implicit Outputs:
0000   149 ;
0000   150 ;       ifb$l_ios       -  i/o status block filled in
0000   151 ;       ( or irb$l_ios if r9 = irab )
0000   152 ;       fwa$q_fib       -  filled in with file function result
0000   153 ;
0000   154 ; Completion Codes:
0000   155 ;
0000   156 ;       qio status code.
0000   157 ;
0000   158 ; Side Effects:
0000   159 ;
0000   160 ;       on return rms will be running at ast level.
0000   161 ;
0000   162 ;--
0000   163
```

```
                     0000   165 ;
                     0000   166 ; r0 contains qio function code
                     0000   167 ; p2-p6 already on stack.
                     0000   168 ; p1 will point to fib descriptor in fwa
                     0000   169 ;
                     0000   170
                     0000   171 RM$FCPFNC::
                     0000   172         $TSTPT  FCPFNC
          54 8ED0    0006   173         POPL    R4                      ; R4 = return pc
        1C AA   7F   0009   174         PUSHAQ  FWA$Q_FIB(R10)          ; p1 = fib descriptor addr
                     000C   175
                     000C   176 ;
                     000C   177 ; r0 contains qio function code
                     000C   178 ; p1 thru p<N> are on stack
                     000C   179 ; finish building qio argument list on stack
                     000C   180 ;
                     000C   181
        51   6E    D0   000C   182 FCPCOM: MOVL    (SP),R1                 ; r1 = fib descriptor address
    51    04 A1    D0   000F   183         MOVL    4(R1),R1                ; r1 = fib address
                     0013   184 FCPCOM_NOFIB:
                     0013   185         ASSUME  QIO$_ASTPRM EQ <QIO$_P1 - 4>
            59   DD   0013   186         PUSHL   R9                      ; astprm = ifab addr
                     0015   187         ASSUME  QIO$_ASTADR EQ <QIO$_ASTPRM -4>
      0000'CF    DF   0015   188         PUSHAL  W^RM$STALLAST           ; ast address
                     0019   189         ASSUME  QIO$_IOSB EQ <QIO$_ASTADR -4>
        0C A9    DF   0019   190         PUSHAL  IFB$L_IOS(R9)           ; i/o status block addr
                     001C   191         ASSUME  QIO$_FUNC EQ <QIO$_IOSB -4>
            50   DD   001C   192         PUSHL   R0                      ; qio function from caller
                     001E   193         ASSUME  QIO$_CHAN EQ <QIO$_FUNC - 4>
        50   59    D0   001E   194         MOVL    R9,R0                   ; set ifab/irab addr
                     0021   195         ASSUME  IFB$B_BID EQ IRB$B_BID
                     0021   196         ASSUME  IFB$V_BID&1 EQ 1
                     0021   197         ASSUME  IRB$V_BID&1 EQ 0
      03 08 A9    E8   0021   198         BLBS    IFB$B_BID(R9),2$        ; branch if ifab
        50   69    D0   0025   199         MOVL    IRB$L_IFAB_LNK(R9),R0   ; get ifab addr
    7E    20 A0    3C   0028   200 2$:     MOVZWL  IFB$W_CHNL(R0),-(SP)    ; i/o channel
            51    D5   002C   201         TSTL    R1                      ; is there a fib?
            05    13   002E   202         BEQL    5$                      ; branch if not
        4F A0    90   0030   203         MOVB    IFB$B_AGENT_MODE(R0),-  ; move agent mode into fib
        2E A1         0033   204                 FIB$B_AGENT_MODE(R1)
                     0035   205         ASSUME  QIO$_EFN EQ <QIO$_CHAN -4>
         FFC8'   30   0035   206 5$:     BSBW    RM$SETEFN               ; set event flag to use
                     0038   207         ASSUME  QIO$_NARGS EQ 12
00000000'9F    0C    FB   0038   208         CALLS   #12,@#SYS$QIO           ; do the qio
        03 50    E9   003F   209         BLBC    R0,10$                  ; branch on error
         FFBB'   30   0042   210         BSBW    RM$STALL                ; await completion
            64   17   0045   211 10$:    JMP     (R4)                    ; return to caller
```

```
                     0047   213 ;
                     0047   214 ; common entry point to perform acp file extend.
                     0047   215 ; (sp)  : return pc
                     0047   216 ; (sp)+4: fib descriptor (quadword)
                     0047   217 ;
                     0047   218
                     0047   219 RM$FCPEXTEND::
                     0047   220         $TSTPT  FCPEXT                  ; count file extends
     54 8ED0         004D   221         POPL    R4                      ; r4 = return pc
50   36   9A         0050   222         MOVZBL  S^#IO$_MODIFY,R0        ; set i/o function code
     7E   7C         0053   223         CLRQ    -(SP)                   ; p6 = p5 = 0
     03   11         0055   224         BRB     PUSH_P4                 ; br to push rest of qio arguments
                     0057   225
                     0057   226 ;
                     0057   227 ; r0 contains qio function code
                     0057   228 ; (sp)  : return pc
                     0057   229 ; (sp)+4: p5
                     0057   230 ; (sp)+8: p6
                     0057   231 ; (sp)+C: fib descriptor (quadword)
                     0057   232 ;
                     0057   233
                     0057   234 RM$FCPFNC_P4::                          ; continue stack at p4
     54 8ED0         0057   235         POPL    R4                      ; r4 = return pc
     7E   7C         005A   236 PUSH_P4:CLRQ    -(SP)                   ; p4 = p3 = 0
     7E   D4         005C   237         CLRL    -(SP)                   ; p2 = 0
  14 AE   9F         005E   238         PUSHAB  20(SP)                  ; p1 = fib descriptor address
     A9   11         0061   239         BRB     FCPCOM                  ; push rest of qio arguments
                     0063   240
                     0063   241 ;
                     0063   242 ; r0 contains qio function code
                     0063   243 ; p1 thru p<N> already on stack
                     0063   244 ;
                     0063   245
                     0063   246 RM$FCPFNC_NOFIB::
     54 8ED0         0063   247         POPL    R4                      ; r4 = return pc
     51   D4         0066   248         CLRL    R1                      ; clear fib address
     A9   11         0068   249         BRB     FCPCOM_NOFIB                    ; push rest of qio arguments
                     006A   250
```

```
006A    252
006A    253          .SBTTL  RM$MAPERR - ROUTINE TO MAP SYSTEM ERROR CODES TO RMS CODES
006A    254
006A    255 ;++
006A    256 ; RM$MAPERR
006A    257 ;
006A    258 ; This routine maps the error code received in r0 to the equivalent rms error
006A    259 ; code.  If the code is not found the default code in r1 is used.  Some system
006A    260 ; errors are mapped to themselves since we don't need special RMS errors for
006A    261 ; them.
006A    262 ;
006A    263 ; Calling sequence:
006A    264 ;
006A    265 ;       brw     rm$maperr
006A    266 ;
006A    267 ; inputs:
006A    268 ;
006A    269 ;       r0      system error code
006A    270 ;       r1      default rms error code
006A    271 ;       r8      fab/rab address
006A    272 ;
006A    273 ; outputs:
006A    274 ;
006A    275 ;       fab$l_stv - set from r0
006A    276 ;       r0      set from mapped error code or r1 if none
006A    277 ;
006A    278 ; Side Effects:
006A    279 ;
006A    280 ;       exits with an rsb.
006A    281 ;--
006A    282
006A    283          .MACRO  MAP SYSCOD, RMSCOD
006A    284          .WORD   SS$_'SYSCOD
006A    285          RMSERR_WORD     RMSCOD
006A    286          .ENDM
006A    287
006A    288 MAPTABLE:
006A    289          MAP     NOSUCHFILE, FNF
006E    290          MAP     NOPRIV, PRV
0072    291          MAP     NOSUCHDEV, DEV
0076    292          MAP     IVDEVNAM, DEV
007A    293          MAP     ENDOFFILE, EOF
007E    294          MAP     DEVALLOC, DNR
0082    295          MAP     DEVNOTMOUNT, DNR
0086    296          MAP     DEVOFFLINE, DNR
008A    297          MAP     DUPFILENAME, FEX
008E    298          MAP     FILNOTEXP, EXP
0092    299          MAP     FCPREADERR, ATR
0096    300          MAP     FCPWRITERR, ATW
009A    301          MAP     ACCONFLICT, FLK
009E    302          MAP     NOMOREFILES, NMF
00A2    303          MAP     DEVICEFULL, FUL
00A6    304          MAP     WRITLCK, WLK
00AA    305          MAP     NOTQUEUED, RLK
00AE    306          MAP     EXENQLM, EXENQLM
00B2    307          MAP     DEADLOCK, DEADLOCK
00B6    308          MAP     INSFMEM, DME
```

```
                 0000 0000   00BA   309          .WORD   0,0                     ; end of table
                             00BE   310
                             00BE   311
                             00BE   312 RM$MAPERR::
          0C A8    50   D0   00BE   313          MOVL    R0,FAB$L_STV(R8)       ; save system code in stv
    50  0FFF0000  8F   D3   00C2   314          BITL    #^X0FFF0000,R0         ; system error code?
                 0E   12   00C9   315          BNEQ    15$                    ; don't map if not
           50  9C  AF   3E   00CB   316          MOVAW   MAPTABLE,R0            ; address of mapping table
          0C A8    80   B1   00CF   317 10$:    CMPW    (R0)+,FAB$L_STV(R8)    ; match with system code?
                 08   13   00D3   318          BEQL    20$                    ; branch if yes
                 80   B5   00D5   319          TSTW    (R0)+                  ; zero entry?
                 F6   12   00D7   320          BNEQ    10$                    ; branch if not
                             00D9   321
                             00D9   322 ;
                             00D9   323 ; reached end of table without finding a match
                             00D9   324 ; use the default code in r1
                             00D9   325 ;
                             00D9   326
             50    51   D0   00D9   327 15$:    MOVL    R1,R0
                   05   00DC   328          RSB
                             00DD   329
                             00DD   330 ;
                             00DD   331 ; found matching code in table - use corresponding rms code
                             00DD   332 ;
                             00DD   333
             50    60   3C   00DD   334 20$:    MOVZWL  (R0),R0
                   05   00E0   335          RSB
                             00E1   336
                             00E1   337          .END
```

```
$$.PSECT_EP                     = 00000000
$$ARGS                          = 0000000C
$$RMSTEST                       = 0000001A
$$RMS_PBUGCHK                   = 00000010
$$RMS_TBUGCHK                   = 00000008
$$RMS_UMODE                     = 00000004
$$T1                            = 00000034
FAB$L_STV                       = 0000000C
FCPCOM                            0000000C R      01
FCPCOM_NOFIB                      00000013 R      01
FIB$B_AGENT_MODE                = 0000002E
FWA$Q_FIB                       = 00000010
IFB$B_AGENT_MODE                = 0000004F
IFB$B_BID                       = 00000008
IFB$C_BID                       = 0000000B
IFB$L_IOS                       = 0000000C
IFB$W_CHNL                      = 00000020
IO$_MODIFY                      = 00000036
IRB$B_BID                       = 00000008
IRB$C_BID                       = 0000000A
IRB$L_IFAB_LNK                  = 00000000
MAPTABLE                          0000006A R      01
PIO$A_TRACE                       ******** X    01
PUSH_P4                           0000005A R      01
QIO$_ASTADR                     = 00000014
QIO$_ASTPRM                     = 00000018
QIO$_CHAN                       = 00000008
QIO$_EFN                        = 00000004
QIO$_FUNC                       = 0000000C
QIO$_IOSB                       = 00000010
QIO$_NARGS                      = 0000000C
QIO$_P1                         = 00000C1C
QIO$_P2                         = 00000020
QIO$_P3                         = 00000024
QIO$_P4                         = 00000028
QIO$_P5                         = 0000002C
QIO$_P6                         = 00000030
RMS$FCPEXTEND                     00000047 RG     01
RMS$FCPFNC                        00000000 RG     01
RMS$FCPFNC_NOFIB                  00000063 RG     01
RMS$FCPFNC_P4                     00000057 RG     01
RMS$MAPERR                        000000BE RG     01
RMS$SETEFN                        ******** X    01
RMS$STALL                         ******** X    01
RMS$STALLAST                      ******** X    01
RMS$_ATR                        = 0001C0CC
RMS$_ATW                        = 0001C0D4
RMS$_DEADLOCK                   = 000187D4
RMS$_DEV                        = 000184C4
RMS$_DME                        = 000184D4
RMS$_DNR                        = 00018272
RMS$_EOF                        = 0001827A
RMS$_EXENQLM                    = 000187DC
RMS$_EXP                        = 000182C2
RMS$_FEX                        = 00018282
RMS$_FLK                        = 0001828A
RMS$_FNF                        = 00018292
RMS$_FUL                        = 00018544
RMS$_NMF                        = 000182CA
RMS$_PRV                        = 0001829A
RMS$_RLK                        = 000182AA
RMS$_WLK                        = 000182BA
SS$_ACCONFLICT                  = 00000800
SS$_DEADLOCK                    = 00000E0A
SS$_DEVALLOC                    = 00000840
SS$_DEVICEFULL                  = 00000850
SS$_DEVNOTMOUNT                 = 0000007C
SS$_DEVOFFLINE                  = 00000084
SS$_DUPFILENAME                 = 00000868
SS$_ENDOFFILE                   = 00000870
SS$_EXENQLM                     = 00002A44
SS$_FCPREADERR                  = 00000888
SS$_FCPWRITERR                  = 000008A0
SS$_FILNOTEXP                   = 000000B4
SS$_INSFMEM                     = 00000124
SS$_IVDEVNAM                    = 00000144
SS$_NOMOREFILES                 = 00000930
SS$_NOPRIV                      = 00000024
SS$_NOSUCHDEV                   = 00000908
SS$_NOSUCHFILE                  = 00000910
SS$_NOTQUEUED                   = 000009B8
SS$_WRITLCK                     = 0000025C
SYS$QIO                           ******** X    01
TPT$L_FCPEXT                      ******** X    01
TPT$L_FCPFNC                      ******** X    01
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
RM$RMS0	000000E1 (225.)	01 (1.)	PIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC BYTE
ABS	00000000 (0.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.11	00:00:00.95
Command processing	115	00:00:00.77	00:00:05.54
Pass 1	458	00:00:17.74	00:00:39.46
Symbol table sort	0	00:00:02.98	00:00:04.97
Pass 2	69	00:00:02.92	00:00:05.22
Symbol table output	11	00:00:00.15	00:00:00.40
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	688	00:00:24.70	00:00:56.56

The working set limit was 1500 pages.
99318 bytes (194 pages) of virtual memory were used to buffer the intermediate code.
There were 110 pages of symbol table space allocated to hold 2065 non-local and 6 local symbols.
337 source lines were read in Pass 1, producing 13 object records in Pass 2.
23 pages of virtual memory were used to define 22 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[RMS.OBJ]RMS.MLB;1	8
_255DUA28:[SYS.OBJ]LIB.MLB;1	0
_255DUA28:[SYSLIB]STARLET.MLB;2	9
TOTALS (all libraries)	17

2170 GETS were required to define 17 macros.

There were no errors, warnings or information messages.

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

RM0EXTRMS
LIS

RM0DIRSCN
LIS

RM0CRECOM
LIS

RM0FABCHK
LIS

RM0CHKSUM
LIS

RM0FWASET
LIS

RM0EXTEND
LIS

RM0IFISI
LIS

RM0JOURNL
LIS

RM0FSETI
LIS

RM0FSET
LIS

RM0COMCLN
LIS

RM0DUMMY
LIS

RM0FILFNC
LIS