```
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
```

FILEID**RMOFWASET

```
RRRRRRRR    MM      MM    000000    FFFFFFFFFF  WW      WW    AAAAAA      SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT
RRRRRRRR    MM      MM    000000    FFFFFFFFFF  WW      WW    AAAAAA      SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT
RR      RR  MMMM  MMMM  00      00  FF          WW      WW   AA    AA  SS          EE              TT
RR      RR  MMMM  MMMM  00      00  FF          WW      WW   AA    AA  SS          EE              TT
RR      RR  MM  MM  MM  00    0000  FF          WW      WW   AA    AA  SS          EE              TT
RR      RR  MM  MM  MM  00    0000  FF          WW      WW   AA    AA  SS          EE              TT
RRRRRRRR    MM      MM  00  00  00  FFFFFFFF    WW      WW   AA    AA    SSSSSS    EEEEEEEE        TT
RRRRRRRR    MM      MM  00  00  00  FFFFFFFF    WW      WW   AA    AA    SSSSSS    EEEEEEEE        TT
RR  RR      MM      MM  0000    00  FF          WW  WW  WW   AAAAAAAAAA        SS  EE              TT
RR  RR      MM      MM  0000    00  FF          WW  WW  WW   AAAAAAAAAA        SS  EE              TT
RR    RR    MM      MM  00      00  FF          WWWW  WWWW   AA    AA        SS  EE              TT
RR    RR    MM      MM  00      00  FF          WWWW  WWWW   AA    AA        SS  EE              TT
RR      RR  MM      MM    000000    FF          WW      WW   AA    AA  SSSSSSSS  EEEEEEEEEE      TT
RR      RR  MM      MM    000000    FF          WW      WW   AA    AA  SSSSSSSS  EEEEEEEEEE      TT

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          $BEGIN  RMOFWASET,000,RM$RMS0,<ALLOCATE AND INITIALIZE FWA>
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     28 ;++
0000     29 ; Facility: RMS32
0000     30 ;
0000     31 ; Abstract:
0000     32 ;
0000     33 ;       This module allocates and initializes a file work
0000     34 ;       area (fwa) page.
0000     35 ;
0000     36 ; Environment:
0000     37 ;               star processor running starlet exec.
0000     38 ;
0000     39 ; Author: L F Laverdure,         creation date: 3-MAR-1977
0000     40 ;
0000     41 ; Modified By:
0000     42 ;
0000     43 ;       V03-019 RAS0321         Ron Schaefer            9-Jul-1984
0000     44 ;               Eliminate access mode itemlist entry for $TRNLNM.
0000     45 ;               Initialize logical name translation mode from FAB.
0000     46 ;
0000     47 ;       V03-018 RAS0296         Ron Schaefer           18-Apr-1984
0000     48 ;               Initialize the secondary device name descriptor.
0000     49 ;
0000     50 ;       V03-017 RAS0298         Ron Schaefer           24-Apr-1984
0000     51 ;               Save/restore FWA$L_ATR_WORK when re-initializing a FWA
0000     52 ;               for searchlists.  Logical complement of JWT0175.
0000     53 ;
0000     54 ;       V03-016 JWT0175         Jim Teague             12-Apr-1984
0000     55 ;               Added extra check when deallocating FWA to see if
0000     56 ;               the ATR work page is still allocated for some reason.
0000     57 ;
0000     58 ;       V03-015 RAS0261         Ron Schaefer            8-Mar-1984
0000     59 ;               Eliminate initialization of FWA$Q_QUOTED as it is
0000     60 ;               now synonymous with FWA$Q_NAME.
0000     61 ;
0000     62 ;       V03-014 RAS0234         Ron Schaefer           11-Jan-1984
0000     63 ;               Add support for SLBHs, revise filename string storage
0000     64 ;               in order to implement multi-line input sticky searchlists.
0000     65 ;
0000     66 ;       V03-013 RAS0226         Ron Schaefer           29-Dec-1983
0000     67 ;               Add FWA $TRNLNM mode processing back in for correct PPF
0000     68 ;               processing.   This was deleted by RAS0219 incorrectly.
0000     69 ;
0000     70 ;       V03-012 RAS0219         Ron Schaefer            8-Dec-1983
0000     71 ;               Reorganize FWA fields and change BID values:
0000     72 ;               Allocate and free new dynamically allocated fields:
0000     73 ;               XLTBUFF1, XLTBUFF2, DN, FN, RN, SWB.
0000     74 ;               Init FWA$T_CDIRxBUF fields for rooted directory names.
0000     75 ;               Init FWA$T_CDEVICEBUF for concealed device name.
0000     76 ;
0000     77 ;       V03-011 SHZ0001         Stephen H. Zalewski    12-Sep-1983
0000     78 ;               Initialize a descriptor to point to the file lock name buffer in
0000     79 ;               the FWA.
0000     80 ;
0000     81 ;       V03-010 KBT0568         Keith B. Thompson      29-Jul-1983
0000     82 ;               Fwa$l_xltsiz changed to a word
0000     83 ;
0000     84 ;       V03-009 KBT0537         Keith B. Thompson      1-Jun-1983
```

```
0000    85 ;                   Do not zero the FN, DN or RN buffers on a search
0000    86 ;                   list operation and set up FWA$L_SLB_PTR correctly
0000    87 ;
0000    88 ;       V03-008 KBT0525         Keith B. Thompson       23-May-1983
0000    89 ;                   Fix a bug
0000    90 ;
0000    91 ;       V03-007 KBT0507         Keith B. Thompson       3-May-1983
0000    92 ;                   Add some new fields to the fwa, add RM$DEALLOCATE_FWA
0000    93 ;                   and put in code to handle search list processing
0000    94 ;
0000    95 ;       V03-006 KBT0458         Keith B. Thompson       10-Jan-1983
0000    96 ;                   Allocate fwa as a real structure
0000    97 ;
0000    98 ;       V03-005 KBT0433         Keith B. Thompson       3-Dec-1982
0000    99 ;                   Init fwa$q_shrfil
0000   100 ;
0000   101 ;       V03-004 KBT0406         Keith B. Thompson       30-Nov-1982
0000   102 ;                   Init fwa$q_dir and fix v3 revision histories
0000   103 ;
0000   104 ;       V03-003 KBT0399         Keith B. Thompson       5-Nov-1982
0000   105 ;                   Forget doing the offset optimazition since the fwa is
0000   106 ;                   beyond all reasonable size
0000   107 ;
0000   108 ;       V03-002 KBT0209         Keith B. Thompson       23-Aug-1982
0000   109 ;                   Reorganize psects
0000   110 ;
0000   111 ;       V03-001 KRM0056         K Malik                 10-Aug-1982
0000   112 ;                   NWA$T_NODEBUF, NWA$T_QUOTEDBUF and NWA$B_UNDER_NOD
0000   113 ;                   were moved to the FWA, so initialize them here.
0000   114 ;                   Also, FWA$B_UNDERLINE changed to FWA$B_UNDER_DEV.
0000   115 ;
0000   116 ;--
```

```
0000   118          .SBTTL  DECLARATIONS
0000   119
0000   120 ;
0000   121 ; Include Files:
0000   122 ;
0000   123
0000   124 ;
0000   125 ; Macros:
0000   126 ;
0000   127
0000   128          $FABDEF
0000   129          $FWADEF
0000   130          $IFBDEF
0000   131          $LNMDEF
0000   132          $PSLDEF
0000   133          $SLBDEF
0000   134          $SLBHDEF
0000   135
0000   136 ;
0000   137 ; Equated Symbols:
0000   138 ;
0000   139
0000   140 ;
0000   141 ; Own Storage:
0000   142 ;
0000   143
```

```
0000    145          .SBTTL  RMSFWASET, Allocate and Initialize FWA
0000    146 ;++
0000    147 ;
0000    148 ;  RMSFWASET    -
0000    149 ;
0000    150 ;       This routine does all kinds of things.
0000    151 ;
0000    152 ;     1. If it is a generic call, i.e. there is no fwa, it calls RMSGETSPC
0000    153 ;        to allocate a fwa and then intializes the various fields
0000    154 ;
0000    155 ;     2. If there is alreay a fwa but this is not a search list operation
0000    156 ;        it allocates and inits a second fwa and chains them together
0000    157 ;
0000    158 ;     3. If there is a fwa and it says it's a search list operation then
0000    159 ;        we simply zero the sucker and continue (saving the SLBH,
0000    160 ;        ATR_WORK and SWB pointers)
0000    161 ;
0000    162 ; Calling sequence:
0000    163 ;
0000    164 ;       BSBW    RMSFWASET
0000    165 ;
0000    166 ; Input Parameters:
0000    167 ;
0000    168 ;       R11     impure area pointer
0000    169 ;       R9      ifab address
0000    170 ;       R8      fab address
0000    171 ;
0000    172 ; Implicit Inputs:
0000    173 ;       none
0000    174 ;
0000    175 ; Output Parameters:
0000    176 ;
0000    177 ;       R10     fwa buffer address
0000    178 ;       R1-R5   destroyed
0000    179 ;       R0      status code
0000    180 ;
0000    181 ; Implicit Outputs:
0000    182 ;
0000    183 ;       fwa buffer descriptors initialized:
0000    184 ;
0000    185 ;               FWA$Q_XLTBUF1
0000    186 ;               FWA$Q_XLTBUF2
0000    187 ;               FWA$Q_SHRFIL
0000    188 ;               FWA$Q_SHRFIL_LCK
0000    189 ;               FWA$Q_AS_SHRFIL
0000    190 ;               FWA$Q_DIR
0000    191 ;               FWA$Q_NODE
0000    192 ;               FWA$Q_QUOTED
0000    193 ;               FWA$Q_DEVICE
0000    194 ;               FWA$Q_DIR1 and all sub directories
0000    195 ;               FWA$Q_NAME
0000    196 ;               FWA$Q_TYPE
0000    197 ;               FWA$Q_VERION
0000    198 ;
0000    199 ;       FWA$T_ITMLST static fields initialized
0000    200 ;       FWA$L_SLBH xLINK queue header initialized
0000    201 ;       FWA$B_UNDER_DEV initialized to the character '_'
```

```
                                0000   202 ;
                                0000   203 ; Completion Codes:
                                0000   204 ;
                                0000   205 ;       SUC or DME
                                0000   206 ;
                                0000   207 ; Side Effects:
                                0000   208 ;
                                0000   209 ;       Uses up tons of memory - have you looked at FWA$C_BLN lately?
                                0000   210 ;
                                0000   211 ;--
                                0000   212
                                0000   213 RM$FWASET::
                5A   38 A9   D0 0000   214         MOVL    IFB$L_FWA_PTR(R9),R10           ; get addr of fwa if any
                       2A   13 0004   215         BEQL    10$                             ; branch if none
                26 6A   02   E1 0006   216         BBC     #FWA$V_SL_PASS,(R10),10$        ; branch if not search list pass
                                000A   217
                                000A   218 ;
                                000A   219 ; We are here for the nth time parsing search list.  Clear the fwa and reinit
                                000A   220 ; it so we can do a clean parse
                                000A   221 ;
                                000A   222
                   58 AA    DD 000A   223         PUSHL   FWA$L_ATR_WORK(R10)             ; save attr scratch page ptr
                   4C AA    DD 000D   224         PUSHL   FWA$L_SWB_PTR(R10)              ; save SWB ptr
             7E    00B0 CA   7D 0010   225         MOVQ    FWA$L_SLBH_FLINK(R10),-(SP)     ; save the SLBH ptrs
6A   093C 8F   00    6E    00 2C 0015   226         MOVC5   #0,(SP),#0,#FWA$C_BLN,(R10)     ; zero the fwa
             00B0 CA    8E   7D 001D   227         MOVQ    (SP)+,FWA$L_SLBH_FLINK(R10)     ; restore SLBH ptrs
                   4C AA 8ED0 0022   228         POPL    FWA$L_SWB_PTR(R10)              ;  and SWB ptr
                   58 AA 8ED0 0026   229         POPL    FWA$L_ATR_WORK(R10)             ;  and attr scratch page ptr
                                002A   230         SSB     #FWA$V_SL_PASS,(R10)            ;  and sl pass flag
                      25   11 002E   231         BRB     INIT_FWA
                                0030   232
                51    59    D0 0030   233 10$:    MOVL    R9,R1                           ; copy address of ifab
52   093C 8F    3C          0033   234         MOVZWL  #FWA$C_BLN,R2                   ; # of bytes
                  FFC5'  30 0038   235         BSBW    RM$GETSPC                       ; allocate FWA => R1
                  01 50   E8 003B   236         BLBS    R0,20$                          ; success continue
                          05 003E   237         RSB                                     ; get out on error
                                003F   238
                                003F   239 ;
                                003F   240 ; If there is already a fwa present ($RENAME) chain them so we can deallocate
                                003F   241 ;
                                003F   242
  48 A1    5A    D0 003F   243 20$:    MOVL    R10,FWA$L_FWA_PTR(R1)           ; chan the fwas
  38 A9    51    D0 0043   244         MOVL    R1,IFB$L_FWA_PTR(R9)            ; stuff the ifab with new fwa
     5A    51    D0 0047   245         MOVL    R1,R10                          ; put fwa in right reg
                                004A   246
50   00B0 CA    DE 004A   247         MOVAL   FWA$L_SLBH_FLINK(R10),R0        ; init SLBH queue
     60    50    D0 004F   248         MOVL    R0,(R0)                         ; fwd link
     60    80    D0 0052   249         MOVL    (R0)+,(R0)                      ; back link
                                0055   250
                                0055   251 ;
                                0055   252 ; Initialize the various fwa buffer descriptors with the size and address
                                0055   253 ; of the related buffer.
                                0055   254 ;
                                0055   255
                                0055   256 INIT_FWA:
  08 AA    28    90 0055   257         MOVB    #FWA$C_BID,FWA$B_BID(R10)       ; set the id
  3C AA    30    3C 0059   258         MOVZWL  #FWA$S_WILD,FWA$Q_DIR(R10)      ; scratch dir buffer
```

```
                40 AA   0868 CA   9E  005D   259          MOVAB   FWA$T_WILD(R10),FWA$Q_DIR+4(R10)
                                    0063   260
                                    0063   261 ;
                                    0063   262 ; Don't need to set the size field. it is set in getdev_char by $getdvi
                                    0063   263 ;
                                    0063   264
            0194 CA   0898 CA   9E  0063   265          MOVAB   FWA$T_SHRFILBUF(R10),FWA$Q_SHRFIL+4(R10)
            019C CA   08A8 CA   9E  006A   266          MOVAB   FWA$T_SHRFIL_LCKNAM(R10),FWA$Q_SHRFIL_LCK+4(R10)
            01A4 CA   08B8 CA   9E  0071   267          MOVAB   FWA$T_AS_SHRFILBUF(R10),FWA$Q_AS_SHRFIL+4(R10)
                                    0078   268
                                    0078   269 ;
                                    0078   270 ; Now all the parsed filename element descriptors (address only)
                                    0078   271 ; (FWA$Q_NODE1-8 descriptors are initialized in RM0XPFN as they are needed)
                                    0078   272 ;
                                    0078   273
            00DC CA   07E9 CA   9E  0078   274          MOVAB   FWA$T_NODEBUF(R10),FWA$Q_NODE+4(R10)    ; Store NODEBUF addr
            00E4 CA   05E9 CA   9E  007E   275          MOVAB   FWA$T_DEVICEBUF(R10),FWA$Q_DEVICE+4(R10)
            00EC CA   06E8 CA   9E  0086   276          MOVAB   FWA$T_CDEVICEBUF(R10),FWA$Q_CONCEAL_DEV+4(R10)
                 55   0246 CA   9E  008D   277          MOVAB   FWA$T_DIR1BUF(R10),R5           ; get address of directory buffers
                 54   0134 CA   9E  0092   278          MOVAB   FWA$Q_DIR1+4(R10),R4            ; get address of directory descr
                 53   037E CA   9E  0097   279          MOVAB   FWA$T_CDIR1BUF(R10),R3          ; get address of root directory buff
                 52   00F4 CA   9E  009C   280          MOVAB   FWA$Q_CDIR1+4(R10),R2           ; get address of directory descr
                           51   D4  00A1   281          CLRL    R1                              ; directories counter
                      64   65   7E  00A3   282 10$:     MOVAQ   (R5),(R4)                       ; store buffer address in descriptor
                      55   27   C0  00A6   283          ADDL2   #FWA$C_DIRBUFSIZ,R5             ; address of next directory buffer
                      54   08   C0  00A9   284          ADDL2   #8,R4                           ; address of next directory descr
                      62   63   7E  00AC   285          MOVAQ   (R3),(R2)                       ; store buffer address in descriptor
                      53   27   C0  00AF   286          ADDL2   #FWA$C_DIRBUFSIZ,R3             ; address of next directory buffer
                      52   08   C0  00B2   287          ADDL2   #8,R2                           ; address of next directory buffer
                   EA 51   07   F3  00B5   288          AOBLEQ  #FWA$C_MAXSUBDIR,R1,10$         ; loop if more
            0174 CA   04B6 CA   9E  00B9   289          MOVAB   FWA$T_NAMEBUF(R10),FWA$Q_NAME+4(R10)
            017C CA   05B6 CA   9E  00C0   290          MOVAB   FWA$T_TYPEBUF(R10),FWA$Q_TYPE+4(R10)
            0184 CA   05DE CA   9E  00C7   291          MOVAB   FWA$T_VERBUF(R10),FWA$Q_VERSION+4(R10)
                                    00CE   292
                                    00CE   293 ;
                                    00CE   294 ; Initalize the static fields of the item list for logical name services
                                    00CE   295 ;
                                    00CE   296
                00010004 8F   D0  00CE   297          MOVL    #<LNM$_INDEX@16>+4,-            ; set index item code
                      5C AA       00D4   298                  FWA$T_ITM_INDEX(R10)            ;  and length
                00030004 8F   D0  00D6   299          MOVL    #<LNM$_ATTRIBUTES@16>+4,-       ; set attributes item code
                      68 AA       00DC   300                  FWA$T_ITM_ATTR(R10)             ;  and length
                000200FF 8F   D0  00DE   301          MOVL    #<LNM$_STRING@16>+255,-         ; set string item code
                      74 AA       00E4   302                  FWA$T_ITM_STRING(R10)           ;  and length
                   009E CA   3E  00E6   303          MOVAW   FWA$W_XLTSIZ(R10),-             ; set result lenght address
                      7C AA       00EA   304                  FWA$T_ITM_STRING+8(R10)
                00070004 8F   D0  00EC   305          MOVL    #<LNM$_MAX_INDEX@16>+4,-        ; set max index item code
                   0080 CA        00F2   306                  FWA$T_ITM_MAX_INDEX(R10)        ;  and length
                                    00F5   307
                                    00F5   308 ;
                                    00F5   309 ; Init the logical name translation access mode field.
                                    00F5   310 ;
                           00   EF  00F5   311          EXTZV   #FAB$V_LNM_MODE,-               ; search name tables desired
                           02       00F7   312                  #FAB$S_LNM_MODE,-               ;
                 50    4A A8       00F8   313                  FAB$B_ACMODES(R8),R0            ;
                           03   12  00FB   314          BNEQ    40$                             ;
                      50   03   9A  00FD   315          MOVZBL  #PSL$C_USER,R0                  ; if none specified, default to user
```

```
                009D CA    50    90  0100   316 40$:    MOVB    R0,FWA$B_XLTMODE(R10)   ; set mode of translation
                                     0105   317
                                     0105   318
                                     0105   319 ;
                                     0105   320 ; Init the 'fake' SLB in the FWA
                                     0105   321 ;
                                     0105   322
                50    00B8 CA    DE  0105   323         MOVAL   FWA$T_SLB(R10),R0               ; get the addr of the fake
                                     010A   324
                                     010A   325         ASSUME  SLB$L_FLINK     EQ      0
                                     010A   326         ASSUME  SLB$L_BLINK     EQ      SLB$L_FLINK+4
                                     010A   327
                      60    50    D0  010A   328         MOVL    R0,(R0)                         ; set up flink
                      60    80    D0  010D   329         MOVL    (R0)+,(R0)                      ; set up blink
                                     0110   330
                                     0110   331 ;
                                     0110   332 ; Move an underscore character to FWA$B_UNDER_DEV & FWA$B_UNDER_NOD
                                     0110   333 ;
                                     0110   334
              05E8 CA    5F 8F    90  0110   335         MOVB    #^A/_/,FWA$B_UNDER_DEV(R10)
              07E8 CA    5F 8F    90  0116   336         MOVB    #^A/_/,FWA$B_UNDER_NOD(R10)
                                     011C   337         RMSSUC
                                05   011F   338         RSB
                                     0120   339
```

```
                     0120   341          .SBTTL  RMSDEALLOCATE_FWA, Deallocate FWA
                     0120   342 ;++
                     0120   343 ;
                     0120   344 ;  RMSDEALLOCATE_FWA -
                     0120   345 ;
                     0120   346 ;       This routine deallocates the fwa(s) and any associated structures
                     0120   347 ;
                     0120   348 ; Calling sequence:
                     0120   349 ;
                     0120   350 ;       BSBW    RMSDEALLOCATE_FWA
                     0120   351 ;
                     0120   352 ; Input Parameters:
                     0120   353 ;
                     0120   354 ;       R11     impure area pointer
                     0120   355 ;       R9      ifab address
                     0120   356 ;
                     0120   357 ; Implicit Inputs:
                     0120   358 ;       none
                     0120   359 ;
                     0120   360 ; Output Parameters:
                     0120   361 ;       none
                     0120   362 ;
                     0120   363 ; Implicit Outputs:
                     0120   364 ;       none
                     0120   365 ;
                     0120   366 ; Completion Codes:
                     0120   367 ;       none
                     0120   368 ;
                     0120   369 ; Side Effects:
                     0120   370 ;
                     0120   371 ;       R0-R5 destroyed
                     0120   372 ;
                     0120   373 ;--
                     0120   374
                     0120   375 RMSDEALLOCATE_FWA::
     7E    56   7D   0120   376          MOVQ    R6,-(SP)                        ; save R6/R7
   57    38 A9   D0   0123   377          MOVL    IFBSL_FWA_PTR(R9),R7            ; get the addr of the fwa
         40   13   0127   378          BEQL    100$                            ; none, then exit
                     0129   379
                     0129   380 ;
                     0129   381 ; We have a fwa, check for SLBs
                     0129   382 ;
                     0129   383
56   00B0 D7   0F   0129   384 10$:    REMQUE  @FWASL_SLBH_FLINK(R7),R6; get slbh list entry
         11   1D   012E   385          BVS     50$                             ; skip if no more
         3B   10   0130   386          BSBB    DEALLOCATE_SLBS
   52    09 A6   9A   0132   387          MOVZBL  SLBHSB_BLN(R6),R2               ; get size of slbh string
      52    14   C0   0136   388          ADDL2   #SLBHSC_BLN,R2                  ; add ovhd
      54    56   D0   0139   389          MOVL    R6,R4                           ; get into right register
          FEC1'  30   013C   390          BSBW    RMSRETSPC1                      ; return slbh
         E8   11   013F   391          BRB     10$                             ; next!
                     0141   392
                     0141   393 ;
                     0141   394 ;  release the SWB if present
                     0141   395 ;
                     0141   396
   54    4C A7   D0   0141   397 50$:    MOVL    FWASL_SWB_PTR(R7),R4            ; get swb ptr
```

```
                03   13  0145   398          BEQL    90$                         ; if there is one
              FEB6'  30  0147   399          BSBW    RM$RETBLK1                  ; return swb
                     0144   400
    54    58 A7    D0  014A   401 90$:       MOVL    FWA$L_ATR_WORK(R7),R4       ; Is the ATR work area allocated?
                03   13  014E   402          BEQL    95$                         ; If not, continue
              FEAD'  30  0150   403          BSBW    RM$RET1PAG                  ; Otherwise, deallocate it
                     0153   404
          48 A7    DD  0153   405 95$:       PUSHL   FWA$L_FWA_PTR(R7)           ; save the address of the next
       54    57    D0  0156   406          MOVL    R7,R4                       ; set fwa addr is proper reg
 '2    093C 8F    3C  0159   407          MOVZWL  #FWA$C_BLN,R2               ; set size of fwa
              FE9F'  30  015E   408          BSBW    RM$RETSPC1                  ; return space
                 57 8ED0  0161   409          POPL    R7                          ; get next fwa
                C3   12  0164   410          BNEQ    10$                         ; branch if there is one
          38 A9    D4  0166   411          CLRL    IFB$L_FWA_PTR(R9)           ; clear the ifab pointer
       56    8E    7D  0169   412 100$:     MOVQ    (SP)+,R6                    ; restore R6/R7
                     05  016C   413          RSB                                 ; exit
                         016D   414
                         016D   415 DEALLOCATE_SLBS:
    54    0C A6    D0  016D   416          MOVL    SLBH$L_SLB_QUE(R6),R4       ; get slb ptr into R4
                0E   13  0171   417          BEQL    20$
                         0173   418
                         0173   419          ASSUME  SLB$L_FLINK     EQ      0
                         0173   420
                64   DD  0173   421 10$:     PUSHL   (R4)                        ; save link
              FE88'  30  0175   422          BSBW    RM$RETBLK1                  ; return slb
                 54 8ED0  0178   423          POPL    R4                          ; retrieve link
 0C A6          54   D1  017B   424          CMPL    R4,SLBH$L_SLB_QUE(R6)       ; was that the last one?
                F2   12  017F   425          BNEQ    10$                         ; no
                     05  0181   426 20$:     RSB                                 ; exit
                         0182   427
                         0182   428          .END
```

```
$$.PSECT_EP                     = 00000000
$$RMSTEST                       = 0000001A
$$RMS_PBUGCHK                   = 00000010
$$RMS_TBUGCHK                   = 00000008
$$RMS_UMODE                     = 00000004
DEALLOCATE_SLBS                   0000016D R      01
FAB$B_ACMODES                   = 0000004A
FAB$S_LNM_MODE                  = 00000002
FAB$V_LNM_MODE                  = 00000000
FWA$B_BID                       = 00000008
FWA$B_UNDER_DEV                 = 000005E8
FWA$B_UNDER_NOD                 = 000007E8
FWA$B_XLTMODE                   = 0000009D
FWA$C_BID                       = 00000028
FWA$C_BLN                       = 0000093C
FWA$C_DIRBUFSIZ                 = 00000027
FWA$C_MAXSUBDIR                 = 00000007
FWA$L_ATR_WORK                  = 00000058
FWA$L_FWA_PTR                   = 00000048
FWA$L_SLB_FLINK                 = 000000B0
FWA$L_SWB_PTR                   = 0000004C
FWA$Q_AS_SHRFIL                 = 000001A0
FWA$Q_CDIR1                     = 000000F0
FWA$Q_CONCEAL_DEV               = 000000E8
FWA$Q_DEVICE                    = 000000E0
FWA$Q_DIR                       = 0000003C
FWA$Q_DIR1                      = 00000130
FWA$Q_NAME                      = 00000170
FWA$Q_NODE                      = 000000D8
FWA$Q_SHRFIL                    = 00000190
FWA$Q_SHRFIL_LCK                = 00000198
FWA$Q_TYPE                      = 00000178
FWA$Q_VERSION                   = 00000180
FWA$S_WILD                      = 00000030
FWA$T_AS_SHRFILBUF              = 000008B8
FWA$T_CDEVICEBUF                = 000006E8
FWA$T_CDIR1BUF                  = 0000037E
FWA$T_DEVICEBUF                 = 000005E9
FWA$T_DIR1BUF                   = 00000246
FWA$T_ITM_ATTR                  = 00000068
FWA$T_ITM_INDEX                 = 0000005C
FWA$T_ITM_MAX_INDEX             = 00000080
FWA$T_ITM_STRING                = 00000074
FWA$T_NAMEBUF                   = 000004B6
FWA$T_NODEBUF                   = 000007E9
FWA$T_SHRFILBUF                 = 00000898
FWA$T_SHRFIL_LCKNAM             = 000008A8
FWA$T_SLB                       = 000000B8
FWA$T_TYPEBUF                   = 000005B6
FWA$T_VERBUF                    = 000005DE
FWA$T_WILD                      = 00000868
FWA$V_SL_PASS                   = 00000002
FWA$W_XLTSIZ                    = 0000009E
IFB$L_FWA_PTR                   = 00000038
INIT_FWA                          00000055 R      01
LNM$_ATTRIBUTES                 = 00000003
LNM$_INDEX                      = 00000001
LNM$_MAX_INDEX                  = 00000007
LNM$_STRING                     = 00000002
PSL$C_USER                      = 00000003
RMS$DEALLOCATE_FWA                00000120 RG     01
RMS$FWASET                        00000000 RG     01
RMS$GETSPC                        ********   X    01
RMS$RETIPAG                       ********   X    01
RMS$RETBLK1                       ********   X    01
RMS$RETSPC1                       ********   X    01
SLB$L_BLINK                     = 00000004
SLB$L_FLINK                     = 00000000
SLBH$B_BLN                      = 00000009
SLBH$C_BLN                      = 00000014
SLBH$L_SLB_QUE                  = 0000000C
```

+----------------+
! Psect synopsis !
+----------------+

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
RMS$RMSO	00000182 (386.)	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	29	00:00:00.08	00:00:01.18
Command processing	116	00:00:00.77	00:00:04.78
Pass 1	267	00:00:07.67	00:00:25.26
Symbol table sort	0	00:00:01.03	00:00:01.92
Pass 2	84	00:00:01.61	00:00:04.36
Symbol table output	10	00:00:00.09	00:00:00.38
Psect synopsis output	1	00:00:00.03	00:00:00.31
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	509	00:00:11.29	00:00:38.19

The working set limit was 1200 pages.
42930 bytes (84 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 781 non-local and 12 local symbols.
428 source lines were read in Pass 1, producing 14 object records in Pass 2.
19 pages of virtual memory were used to define 18 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	6
TOTALS (all libraries)	14

883 GETS were required to define 14 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:RMOFWASET/OBJ=OBJ$:RMOFWASET MSRC$:RMOFWASET/UPDATE=(ENH$:RMOFWASET)+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