```
FFFFFFFFFFFFFFF        111         111    XXX       XXX
FFFFFFFFFFFFFFF        111         111    XXX       XXX
FFFFFFFFFFFFFFF        111         111    XXX       XXX
FFF                 111111      111111    XXX       XXX
FFF                 111111      111111    XXX       XXX
FFF                 111111      111111    XXX       XXX
FFF                    111         111       XXX XXX
FFF                    111         111       XXX XXX
FFF                    111         111       XXX XXX
FFFFFFFF.FFF           111         111          XXX
FFFFFFFFFFFF           111         111          XXX
FFFFFFFFFFFF           111         111          XXX
FFF                    111         111       XXX XXX
FFF                    111         111       XXX XXX
FFF                    111         111       XXX XXX
FFF                    111         111    XXX       XXX
FFF                    111         111    XXX       XXX
FFF                    111         111    XXX       XXX
FFF                 111111111   111111111 XXX       XXX
FFF                 111111111   111111111 XXX       XXX
FFF                 111111111   111111111 XXX       XXX
```

```
_$25
Symb
----
IOC
IO_C
IO_C
IO_D
IO_F
IO_S
KILL

KILL
KILL
LB_E
LB_D
LB_F
LB_H
LB_L
LOCA
LOCA
LOCK

LOCK
LOCK
LOCK
LOC_
LOC_
L_CO
L_CO
L_DA
L_DA
MAIN
MAKE
MAKE
MAKE
MAKE
MAKE
MAKE

MAKE
MAKE
MAP_
MAP_

MAP_
MARK
MARK
MARK
MARK
MARK
```

```
DDDDDDDD    EEEEEEEEEE  LL          EEEEEEEEEE  TTTTTTTTTT  EEEEEEEEEE
DDDDDDDD    EEEEEEEEEE  LL          EEEEEEEEEE  TTTTTTTTTT  EEEEEEEEEE
DD      DD  EE          LL          EE              TT      EE
DD      DD  EE          LL          EE              TT      EE
DD      DD  EE          LL          EE              TT      EE
DD      DD  EE          LL          EE              TT      EE
DD      DD  EEEEEEEE    LL          EEEEEEEE        TT      EEEEEEEE
DD      DD  EEEEEEEE    LL          EEEEEEEE        TT      EEEEEEEE
DD      DD  EE          LL          EE              TT      EE
DD      DD  EE          LL          EE              TT      EE
DD      DD  EE          LL          EE              TT      EE              ....
DD      DD  EE          LL          EE              TT      EE              ....
DDDDDDDD    EEEEEEEEEE  LLLLLLLLLL  EEEEEEEEEE      TT      EEEEEEEEEE      ....
DDDDDDDD    EEEEEEEEEE  LLLLLLLLLL  EEEEEEEEEE      TT      EEEEEEEEEE      ....


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
```

```
;     1          0001  0 MODULE DELETE (
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 IDENT = 'V04-000'
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1
;     7          0007  1 !
;     8          0008  1 !*****************************************************************************
;     9          0009  1 !*                                                                           *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13          0013  1 !*                                                                           *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    19          0019  1 !*  TRANSFERRED.                                                              *
;    20          0020  1 !*                                                                           *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    23          0023  1 !*  CORPORATION.                                                              *
;    24          0024  1 !*                                                                           *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*****************************************************************************
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 !
;    33          0033  1 ! FACILITY:  F11ACP Structure Level 2
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       This routine performs the DELETE function.
;    38          0038  1 !
;    39          0039  1 ! ENVIRONMENT:
;    40          0040  1 !
;    41          0041  1 !       STARLET operating system, including privileged system services
;    42          0042  1 !       and internal exec routines.
;    43          0043  1 !
;    44          0044  1 !--
;    45          0045  1 !
;    46          0046  1 !
;    47          0047  1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  1-Apr-1977
;    48          0048  1 !
;    49          0049  1 ! MODIFIED BY:
;    50          0050  1 !
;    51          0051  1 !       V03-024 CDS0015         Christian D. Saether    14-Aug-1984
;    52          0052  1 !               Modify handling of extension fcbs.
;    53          0053  1 !
;    54          0054  1 !       V03-023 CDS0014         Christian D. Saether    10-Aug-1984
;    55          0055  1 !               Clear directory flag in header prior to actually
;    56          0056  1 !               deleting file so that extra checks against deleting
;    57          0057  1 !               a directory can be made in delete_file.
```

```
;  58          0058  1 ;
;  59          0059  1 ;       V03-022 CDS0013         Christian D. Saether     7-Aug-1984
;  60          0060  1 ;               Wipe out directory index if there is one when
;  61          0061  1 ;               deleting the fcb.  Use common routine to delete fcb.
;  62          0062  1 ;
;  63          0063  1 ;       V03-021 CDS0012         Christian D. Saether     6-Aug-1984
;  64          0064  1 ;               Sense of test in CDS0011 to fix access arbitration
;  65          0065  1 ;               on exclusively accessed file was wrong.  Fix it.
;  66          0066  1 ;
;  67          0067  1 ;       V03-020 CDS0011         Christian D. Saether    31-July-1984
;  68          0068  1 ;               Remove local declaration of get_map_pointer linkage.
;  69          0069  1 ;               Fix access arbitration check to allow deletion if
;  70          0070  1 ;               we have it accessed exclusively readonly.
;  71          0071  1 ;
;  72          0072  1 ;       V03-019 LMP0275         L. Mark Pilant,          23-Jul-1984  14:19
;  73          0073  1 ;               Don't try to delete an uninitialized ACL.
;  74          0074  1 ;
;  75          0075  1 ;       V03-018 ACG0427         Andrew C. Goldstein,     8-May-1984  13:32
;  76          0076  1 ;               Write audit record for file about to be deleted
;  77          0077  1 ;
;  78          0078  1 ;       V03-017 CDS0010         Christian D. Saether     4-May-1984
;  79          0079  1 ;               Remember to release access lock in MARKDEL_FCB if
;  80          0080  1 ;               we get rid of the fcb there.
;  81          0081  1 ;
;  82          0082  1 ;       V03-016 CDS0009         Christian D. Saether     19-Apr-1984
;  83          0083  1 ;               Changes to restore compatible (with V3) delete behavior.
;  84          0084  1 ;
;  85          0085  1 ;       V03-015 ACG0415         Andrew C. Goldstein,     5-Apr-1984  21:31
;  86          0086  1 ;               Interface change to ACL_DELETEACL
;  87          0087  1 ;
;  88          0088  1 ;       V03-014 ACG0412         Andrew C. Goldstein,     22-Mar-1984  18:21
;  89          0089  1 ;               Implement agent access mode support; add access mode to
;  90          0090  1 ;               check protection call
;  91          0091  1 ;
;  92          0092  1 ;       V03-013 ACG0408         Andrew C. Goldstein,     20-Mar-1984  17:35
;  93          0093  1 ;               Make APPLY_RVN and DEFAULT_RVN macros; remove delete logger
;  94          0094  1 ;
;  95          0095  1 ;       V03-012 CDS0008         Christian D. Saether     23-Feb-1984
;  96          0096  1 ;               Change references to FLUSH_LOCK_BASIS to WRITE_DIRTY.
;  97          0097  1 ;               Checksum header and mark dirty when only marking
;  98          0098  1 ;               for delete and not actually deleting file.
;  99          0099  1 ;               Modify call to ACL_DELETEACL.
; 100          0100  1 ;
; 101          0101  1 ;       V03-011 CDS0007         Christian D. Saether     17-Jan-1984
; 102          0102  1 ;               Modify interface to APPLY_RVN.
; 103          0103  1 ;
; 104          0104  1 ;       V03-010 CDS0006         Christian D. Saether     27-Dec-1983
; 105          0105  1 ;               Use BIND_COMMON macro.
; 106          0106  1 ;
; 107          0107  1 ;       V03-009 CDS0005         Christian D. Saether     13-Dec-1983
; 108          0108  1 ;               Move all OWN data declarations to the
; 109          0109  1 ;               COMMON module.
; 110          0110  1 ;
; 111          0111  1 ;       V03-008 LMP0178         L. Mark Pilant,          8-Dec-1983  14:22
; 112          0112  1 ;               Fix a bug that caused paged pool to be lost when deleting
; 113          0113  1 ;               an unaccessed file.
; 114          0114  1 ;
```

```
;  115     0115  1 !       V03-007 ACG0368         Andrew C. Goldstein,    4-Nov-1983  14:24
;  116     0116  1 !               Handle short ident areas in back link file name check
;  117     0117  1 !
;  118     0118  1 !       V03-006 CDS0004         Christian D. Saether    14-Sep-1983
;  119     0119  1 !               Modify SERIAL_FILE interface.
;  120     0120  1 !               Call RELEASE_SERIAL_LOCK to dequeue.
;  121     0121  1 !
;  122     0122  1 !       V03-005 CDS0003         Christian D. Saether     6-May-1983
;  123     0123  1 !               Call SERIAL_FILE to interlock file processing.
;  124     0124  1 !               Remove SWITCH_VOLUME and SEARCH_FCB calls in DELETE
;  125     0125  1 !               routine because they are called from MARK_DELETE now.
;  126     0126  1 !               Call FLUSH_FID at the end of MARK_DELETE so that
;  127     0127  1 !               file processing interlock can be released.  This is
;  128     0128  1 !               necessary because of the call from CREATE using
;  129     0129  1 !               secondary context.
;  130     0130  1 !
;  131     0131  1 !       V03-004 ACG0323         Andrew C. Goldstein,    12-Apr-1983  16:12
;  132     0132  1 !               Fix passing of result string buffer
;  133     0133  1 !
;  134     0134  1 !       V03-003 CDS0002         Christian D. Saether     7-Apr-1983
;  135     0135  1 !               Modifications to correctly arbitrate delete actions
;  136     0136  1 !               in a cluster.
;  137     0137  1 !
;  138     0138  1 !       V03-002 ACG0323         Andrew C. Goldstein,    25-Mar-1983  16:29
;  139     0139  1 !               Erase back link when matching directory entry is removed
;  140     0140  1 !
;  141     0141  1 !       V03-001 LMP0059         L. Mark Pilant,         27-Dec-1982  8:14
;  142     0142  1 !               Always create an FCB for a file header.  This eliminates a
;  143     0143  1 !               lot of special case FCB handling.
;  144     0144  1 !
;  145     0145  1 !       V02-006 ACG0249         Andrew C. Goldstein,    29-Dec-1981  13:58
;  146     0146  1 !               Use DATA block type to read directory block
;  147     0147  1 !
;  148     0148  1 !       V02-005 ACG0227         Andrew C. Goldstein,    24-Nov-1981  22:45
;  149     0149  1 !               Protect directory files from deletion
;  150     0150  1 !
;  151     0151  1 !       V02-004 ACG0167         Andrew C. Goldstein,    16-Apr-1980  19:25
;  152     0152  1 !               Previous revision history moved to F11B.REV
;  153     0153  1 !**
;  154     0154  1
;  155     0155  1
;  156     0156  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;  157     0157  1 REQUIRE 'SRC$:FCPDEF.B32';
;  158     1148  1
;  159     1149  1
;  160     1150  1 FORWARD ROUTINE
;  161     1151  1         DELETE          : L_NORM,       ! main delete function
;  162     1152  1         MARK_DELETE     : L_NORM NOVALUE, ! mark file for delete
;  163     1153  1         MARKDEL_FCB     : L_NORM,       ! mark FCB of file for delete
;  164     1154  1         DELETE_HANDLER  : L_NORM;       ! condition handler for delete function
```

```
;  166     1155  1 GLOBAL ROUTINE DELETE : L_NORM =
;  167     1156  1
;  168     1157  1 !++
;  169     1158  1 !
;  170     1159  1 ! FUNCTIONAL DESCRIPTION:
;  171     1160  1 !
;  172     1161  1 !       This routine performs the remove and mark for delete functions.
;  173     1162  1 !
;  174     1163  1 ! CALLING SEQUENCE:
;  175     1164  1 !       DELETE ()
;  176     1165  1 !
;  177     1166  1 ! INPUT PARAMETERS:
;  178     1167  1 !       NONE
;  179     1168  1 !
;  180     1169  1 ! IMPLICIT INPUTS:
;  181     1170  1 !       IO_PACKET: I/O packet in process
;  182     1171  1 !
;  183     1172  1 ! OUTPUT PARAMETERS:
;  184     1173  1 !       PRIMARY_FCB: FCB of file
;  185     1174  1 !
;  186     1175  1 ! IMPLICIT OUTPUTS:
;  187     1176  1 !       NONE
;  188     1177  1 !
;  189     1178  1 ! ROUTINE VALUE:
;  190     1179  1 !       1
;  191     1180  1 !
;  192     1181  1 ! SIDE EFFECTS:
;  193     1182  1 !       directory entry removed
;  194     1183  1 !       file marked for delete or deleted
;  195     1184  1 !
;  196     1185  1 !--
;  197     1186  1
;  198     1187  2 BEGIN
;  199     1188  2
;  200     1189  2 LOCAL
;  201     1190  2         ABD             : REF BBLOCKVECTOR [,ABD$C_LENGTH],
;  202     1191  2                                           ! buffer descriptors
;  203     1192  2         FIB             : REF BBLOCK,     ! FIB
;  204     1193  2         RESULT_LENGTH,                    ! length of name string from directory
;  205     1194  2         RESULT          : VECTOR [FILENAME_LENGTH+6, BYTE];
;  206     1195  2                                           ! file name string from directory
;  207     1196  2
;  208     1197  2 BIND_COMMON;
;  209     1198  2
;  210     1199  2 EXTERNAL ROUTINE
;  211     1200  2         GET_FIB         : L_NORM,         ! get FIB of request
;  212     1201  2         FIND            : L_NORM;         ! find name in directory
;  213     1202  2
;  214     1203  2
;  215     1204  2 ! First find the buffer descriptor, FIB, FCB, etc. then remove the
;  216     1205  2 ! directory entry.
;  217     1206  2 !
;  218     1207  2
;  219     1208  2                                           ! pointer to buffer descriptors
;  220     1209  2 ABD = .BBLOCK [.IO_PACKET[IRP$L_SVAPTE], AIB$L_DESCRIPT];
;  221     1210  2 FIB = GET_FIB (.ABD);
;  222     1211  2
```

```
;  223          1212  2 IF .CURRENT_VCB[VCB$V_NOALLOC]
;  224          1213  2 THEN ERR_EXIT (SS$_WRITLCK);
;  225          1214  2
;  226          1215  2 ! If a directory ID is present, do a directory search first and remove
;  227          1216  2 ! the directory entry.
;  228          1217  2 !
;  229          1218  2
;  230          1219  2 RESULT_LENGTH = 0;
;  231          1220  2 IF .CLEANUP_FLAGS[CLF_DIRECTORY]
;  232          1221  2 THEN FIND (.ABD, .FIB, 1, RESULT_LENGTH, RESULT);
;  233          1222  2
;  234          1223  2 ! If there is a file open on the channel, check the file ID returned by the
;  235          1224  2 ! FIND against that of the open file. If they do not match, treat the file
;  236          1225  2 ! as if it were not open.
;  237          1226  2 !
;  238          1227  2
;  239          1228  2 IF .PRIMARY_FCB NEQ 0
;  240          1229  2 THEN
;  241          1230  3     BEGIN
;  242          1231  3     IF .PRIMARY_FCB[FCB$W_FID_NUM] NEQ .FIB[FIB$W_FID_NUM]
;  243          1232  3     OR .PRIMARY_FCB[FCB$W_FID_SEQ] NEQ .FIB[FIB$W_FID_SEQ]
;  244          1233  3     THEN CURRENT_WINDOW = 0;
;  245          1234  2     END;
;  246          1235  2
;  247          1236  2 ! Now actually mark the file for delete if requested.
;  248          1237  2 !
;  249          1238  2
;  250          1239  2 MARK_DELETE (.FIB, .BBLOCK [IO_PACKET[IRP$W_FUNC], IO$V_DELETE], .RESULT_LENGTH, RESULT);
;  251          1240  2
;  252          1241  2 RETURN 1;
;  253          1242  2
;  254          1243  1 END;                                     ! end of routine DELETE

                                                        .TITLE  DELETE
                                                        .IDENT  \V04-000\

                                                        .EXTRN  GET_FIB, FIND

                                                        .PSECT  $CODE$,NOWRT,2

                                        000C 00000      .ENTRY  DELETE, Save R2,R3                      ;  1155
                              5E    A4  AE  9E 00002      MOVAB   -92(SP), SP                             ;
                              50    90  AA  D0 00006      MOVL    -112(BASE), R0                          ;  1209
                              53    2C  B0  D0 0000A      MOVL    @44(R0), ABD                            ;
                                        53  DD 0000E      PUSHL   ABD                                     ;  1210
                        0000G CF        01  FB 00010      CALLS   #1, GET_FIB                             ;
                              52        50  D0 00015      MOVL    R0, FIB                                 ;
                              50    98  AA  D0 00018      MOVL    -104(BASE), R0                          ;  1212
                     05    0B   A0        04  E1 0001C      BBC     #4, 11(R0), 1$                          ;
                                  025C  8F  BF 00021      CHMU    #604                                    ;  1213
                                        04  00025      RET                                             ;
                                        6E  D4 00026 1$:  CLRL    RESULT_LENGTH                         ;  1219
                     11               6A  06  E1 00028      BBC     #6, (BASE), 2$                          ;  1220
                                  04    AE  9F 0002C      PUSHAB  RESULT                                  ;  1221
                                  04    AE  9F 0002F      PUSHAB  RESULT_LENGTH                           ;
                                        01  DD 00032      PUSHL   #1                                      ;
```

```
                                                 52  DD 00034          PUSHL   FIB                                                    ;
                                                 53  DD 00036          PUSHL   ABD                                                    ;
                                0000G    CF      05  FB 00038          CALLS   #5, FIND                                               ;
                                         50  08  AA  D0 0003D 2$:      MOVL    8(BASE), R0                                            ; 1228
                                             11  13  00041            BEQL    4$                                                     ;
                                   04    A2  24  A0  B1 00043          CMPW    36(R0), 4(FIB)                                         ; 1231
                                             07  12  00048            BNEQ    3$                                                     ;
                                   06    A2  26  A0  B1 0004A          CMPW    38(R0), 6(FIB)                                         ; 1232
                                             03  13  0004F            BEQL    4$                                                     ;
                                             0C  AA  D4 00051 3$:      CLRL    12(BASE)                                               ; 1233
                                             04  AE  9F 00054 4$:      PUSHAB  RESULT                                                 ; 1239
                                             04  AE  DD 00057          PUSHL   RESULT_LENGTH                                          ;
                                         50  90  AA  D0 0005A          MOVL    -112(BASE), R0                                         ;
                7E      21    A0             01  00  EF 0005E          EXTZV   #0, #1, 33(R0), -(SP)                                  ;
                                                 52  DD 00064          PUSHL   FIB                                                    ;
                                0000V    CF      04  FB 00066          CALLS   #4, MARK_DELETE                                        ;
                                         50      01  D0 0006B          MOVL    #1, R0                                                 ; 1241
                                                 04  00070            RET                                                            ; 1243

; Routine Size:  111 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  256          1244  1 GLOBAL ROUTINE MARK_DELETE (FIB, DO_DELETE, RESULT_LENGTH, RESULT) : L_NORM NOVALUE =
;  257          1245  1
;  258          1246  1 !++
;  259          1247  1 !
;  260          1248  1 ! FUNCTIONAL DESCRIPTION:
;  261          1249  1 !
;  262          1250  1 !       This routine marks the indicated file for delete and deletes it
;  263          1251  1 !       if it is not accessed.
;  264          1252  1 !
;  265          1253  1 ! CALLING SEQUENCE:
;  266          1254  1 !       MARK_DELETE (ARG1, ARG2, ARG3, ARG4)
;  267          1255  1 !
;  268          1256  1 ! INPUT PARAMETERS:
;  269          1257  1 !       ARG1: address of FIB
;  270          1258  1 !       ARG2: 1 to actually delete the file
;  271          1259  1 !             0 to only remove the directory entry
;  272          1260  1 !       ARG3: length of name string from directory operation
;  273          1261  1 !       ARG4: address of name string
;  274          1262  1 !
;  275          1263  1 ! IMPLICIT INPUTS:
;  276          1264  1 !       NONE
;  277          1265  1 !
;  278          1266  1 ! OUTPUT PARAMETERS:
;  279          1267  1 !       NONE
;  280          1268  1 !
;  281          1269  1 ! IMPLICIT OUTPUTS:
;  282          1270  1 !       NONE
;  283          1271  1 !
;  284          1272  1 ! ROUTINE VALUE:
;  285          1273  1 !       NONE
;  286          1274  1 !
;  287          1275  1 ! SIDE EFFECTS:
;  288          1276  1 !       file marked for delete or deleted
;  289          1277  1 !
;  290          1278  1 !--
;  291          1279  1
;  292          1280  2 BEGIN
;  293          1281  2
;  294          1282  2 BUILTIN
;  295          1283  2         FP;
;  296          1284  2
;  297          1285  2 MAP
;  298          1286  2         FIB             : REF BBLOCK;   ! FIB
;  299          1287  2
;  300          1288  2 GLOBAL REGISTER
;  301          1289  2         COUNT           = 6,            ! map pointer count
;  302          1290  2         LBN             = 7,            ! map pointer LBN
;  303          1291  2         MAP_POINTER     = 8;            ! pointer to file header map area
;  304          1292  2
;  305          1293  2 LOCAL
;  306          1294  2         CURR_LKMODE,                    ! mode access lock currently held at.
;  307          1295  2         EOF,                            ! end of file VBN of file
;  308          1296  2         BUFFER          : REF VECTOR [,WORD], ! buffer address of block read
;  309          1297  2         FCB             : REF BBLOCK,   ! FCB of file
;  310          1298  2         HEADER          : REF BBLOCK,   ! file header
;  311          1299  2         IDENT_AREA      : REF BBLOCK,   ! header's ident area
;  312          1300  2         TEMP_FID        : BBLOCK [FID$C_LENGTH], ! temp copy of file ID
```

```
;  313          1301  2          FCB_CREATED,                    ! Flag indicating new FCB created
;  314          1302  2          NEW_HEADER      : REF BBLOCK,   ! Address of extension header
;  315          1303  2          ARGLIST         : REF BBLOCK;   ! pointer to audit block entries
;  316          1304  2
;  317          1305  2 BIND_COMMON;
;  318          1306  2
;  319          1307  2 EXTERNAL ROUTINE
;  320          1308  2          REBLD_PRIM_FCB  : L_NORM NOVALUE, ! rebuild primary fcb from header
;  321          1309  2          BUILD_EXT_FCBS  : L_NORM NOVALUE, ! build extension fcb chain
;  322          1310  2          KILL_DIRX       : L_NORM NOVALUE, ! delete directory index
;  323          1311  2          KILL_BUFFERS    : L_NORM NOVALUE, ! kill directory buffers
;  324          1312  2          NUKE_HEAD_FCB   : L_NORM NOVALUE, ! cleanup and delete prim fcb
;  325          1313  2          DEL_EXTFCB      : L_NORM,         ! delete extension FCBs.
;  326          1314  2          ARBITRATE_ACCESS : L_JSB_2ARGS,   ! determine allowed file access
;  327          1315  2          CONV_ACCLOCK    : L_NORM,         ! convert file access lock.
;  328          1316  2          WRITE_DIRTY     : L_NORM,         ! write back modified buffers.
;  329          1317  2          SERIAL_FILE     : L_NORM,         ! serialize file processing
;  330          1318  2          RELEASE_SERIAL_LOCK : L_NORM NOVALUE,
;  331          1319  2          SWITCH_VOLUME   : L_NORM,         ! switch context to desired volume
;  332          1320  2          SEARCH_FCB      : L_NORM,         ! search FCB list
;  333          1321  2          CREATE_FCB      : L_NORM,         ! create an FCB
;  334          1322  2          READ_HEADER     : L_NORM,         ! read file header
;  335          1323  2          CHECK_PROTECT   : L_NORM,         ! check file protection
;  336          1324  2          WRITE_AUDIT     : L_NORM,         ! write audit record
;  337          1325  2          GET_MAP_POINTER : L_MAP_POINTER, ! get file header map pointer
;  338          1326  2          READ_BLOCK      : L_NORM,         ! read a disk block
;  339          1327  2          INVALIDATE      : L_NORM,         ! invalidate block buffer
;  340          1328  2          MARK_DIRTY      : L_NORM,         ! mark buffer for write-back
;  341          1329  2          DELETE_FILE     : L_NORM,         ! delete the file
;  342          1330  2          CHECKSUM        : L_NORM;         ! checksum file header
;  343          1331  2
;  344          1332  2
;  345          1333  2 ! Find the FCB, if any, and then read the header. Reading the header is done
;  346          1334  2 ! under a condition handler that quietly exits with success if errors are
;  347          1335  2 ! encountered. Thus, deleting a bad file header succeeds quietly.
;  348          1336  2 !
;  349          1337  2
;  350          1338  2 SWITCH_VOLUME (.FIB[FIB$W_FID_RVN]);
;  351          1339  2
;  352          1340  2 ! Serialize further processing on this file.
;  353          1341  2 !
;  354          1342  2
;  355          1343  2 PRIM_LCKINDX = SERIAL_FILE (FIB [FIB$W_FID]);
;  356          1344  2
;  357          1345  2 FCB = SEARCH_FCB (FIB[FIB$W_FID]);
;  358          1346  2 SAVE_STATUS = .USER_STATUS;
;  359          1347  2 .FP = DELETE_HANDLER;
;  360          1348  2 HEADER = READ_HEADER (FIB[FIB$W_FID], .FCB);
;  361          1349  2 .FP = 0;
;  362          1350  2
;  363          1351  2 ! If this is a real delete, proceed with it.
;  364          1352  2 !
;  365          1353  2
;  366          1354  2 IF .DO_DELETE
;  367          1355  2 THEN
;  368          1356  3     BEGIN
;  369          1357  3
```

```
;  370          1358  3 ! Check that the file is not a reserved file (FID less than
;  371          1359  3 ! .CURRENT_VCB[VCB$B_RESFILES]).
;  372          1360  3 !
;  373          1361  3
;  374          1362  3     IF .FIB[FIB$W_FID_NUM] LEQU .CURRENT_VCB[VCB$B_RESFILES]
;  375          1363  3     AND .FIB[FIB$B_FID_NMX] EQL 0
;  376          1364  3     THEN ERR_EXIT (SS$_NOPRIV);
;  377          1365  3
;  378          1366  3 ! At this point, build the necessary FCB chain to allow the ACL to be built.
;  379          1367  3
;  380          1368  3     FCB_CREATED = 0;
;  381          1369  3     IF .FCB EQL 0
;  382          1370  3     THEN
;  383          1371  4         BEGIN
;  384          1372  4         FCB_CREATED = 1;
;  385          1373  4         FCB = KERNEL_CALL (CREATE_FCB, .HEADER);
;  386          1374  3         END;
;  387          1375  3     PRIMARY_FCB = .FCB;                         ! Record FCB for external use
;  388          1376  3
;  389          1377  3 ! If the file is multi-header, read in the extension headers and create
;  390          1378  3 ! extension FCB's.  Finally, read back the primary header.
;  391          1379  3 !
;  392          1380  3
;  393          1381  3     IF .FCB_CREATED
;  394          1382  3     THEN
;  395          1383  3         BUILD_EXT_FCBS (.HEADER)
;  396          1384  3     ELSE
;  397          1385  3         IF .FCB [FCB$V_STALE]
;  398          1386  3         THEN
;  399          1387  4             BEGIN
;  400          1388  4
;  401          1389  4             REBLD_PRIM_FCB (.FCB, .HEADER);
;  402          1390  4
;  403          1391  4             BUILD_EXT_FCBS (.HEADER);
;  404          1392  4
;  405          1393  3             END;
;  406          1394  3
;  407          1395  3 ! Check file protection. Check if the file is write accessed by someone
;  408          1396  3 ! else and not the deleter.
;  409          1397  3 !
;  410          1398  3
;  411          1399  3     CHECK_PROTECT (DELETE_ACCESS, .HEADER, .FCB,
;  412          1400  3                    MAXU (.IO_PACKET[IRP$V_MODE], .FIB[FIB$B_AGENT_MODE]));
;  413          1401  3
;  414          1402  3 ! If the file is identified as a directory, check to see if it is empty.
;  415          1403  3 ! Non-empty directories cannot be deleted under any circumstances.
;  416          1404  3 ! The check for emptiness is done by (1) checking for a length of
;  417          1405  3 ! 1 block, and (2) reading the block and looking for the data pattern of
;  418          1406  3 ! an empty directory block.
;  419          1407  3 !
;  420          1408  3
;  421          1409  3     IF .HEADER[FH2$V_DIRECTORY]
;  422          1410  3     THEN
;  423          1411  4         BEGIN
;  424          1412  4         EOF = ROT (.BBLOCK [HEADER[FH2$W_RECATTR], FAT$L_EFBLK], 16);
;  425          1413  4         IF .EOF NEQ 0
;  426          1414  4         AND .BBLOCK [HEADER[FH2$W_RECATTR], FAT$W_FFBYTE] EQL 0
```

```
;  427          1415  4          THEN EOF = .EOF - 1;
;  428          1416  4          IF .EOF LEQU 1
;  429          1417  4          THEN
;  430          1418  5              BEGIN
;  431          1419  5              MAP_POINTER = .HEADER + .HEADER[FH2$B_MPOFFSET] * 2;
;  432          1420  5              GET_MAP_POINTER ();
;  433          1421  5              BUFFER = READ_BLOCK (.LBN, 1 , DATA_TYPE);
;  434          1422  5              IF .BUFFER[0] NEQ 65535
;  435          1423  5              THEN ERR_EXIT (SS$_DIRNOTEMPTY);
;  436          1424  5              INVALIDATE (.BUFFER);
;  437          1425  5              END
;  438          1426  4          ELSE ERR_EXIT (SS$_DIRNOTEMPTY);
;  439          1427  4
;  440          1428  3          END;
;  441          1429  3
;  442          1430  3  ! Check if a security audit record is to be written for this file.
;  443          1431  3  ! If so, now is the last time to do it. ("Morituri te salutamus!")
;  444          1432  3  !
;  445          1433  3
;  446          1434  3      ARGLIST = AUDIT_ARGLIST;
;  447          1435  3      DECR J FROM MAX_AUDIT_COUNT TO 1
;  448          1436  3      DO
;  449          1437  4          BEGIN
;  450          1438  4          IF .ARGLIST[AUDIT_TYPE] NEQ 0
;  451          1439  4          AND .BBLOCK [ARGLIST[AUDIT_FID], FID$W_NUM] EQL .FCB[FCB$W_FID_NUM]
;  452          1440  4          AND .BBLOCK [ARGLIST[AUDIT_FID], FID$W_SEQ] EQL .FCB[FCB$W_FID_SEQ]
;  453          1441  4          AND .BBLOCK [ARGLIST[AUDIT_FID], FID$W_RVN] EQL .FCB[FCB$W_FID_RVN]
;  454          1442  4          THEN
;  455          1443  5              BEGIN
;  456          1444  5              WRITE_AUDIT (.ARGLIST);
;  457          1445  5              HEADER = .FILE_HEADER;
;  458          1446  5              EXITLOOP 0;
;  459          1447  4              END;
;  460          1448  4          ARGLIST = .ARGLIST + AUDIT_LENGTH;
;  461          1449  3          END;
;  462          1450  3
;  463          1451  3  ! Remember current lock mode to be restored later, if necessary.
;  464          1452  3  !
;  465          1453  3
;  466          1454  3      CURR_LKMODE = .FCB [FCB$B_ACCLKMODE];
;  467          1455  3
;  468          1456  3  ! Make access checks.
;  469          1457  3  ! If we have the file accessed, we may delete it as long as we have
;  470          1458  3  ! write access ourselves (whether there are other writers or not).
;  471          1459  3  ! In all other cases, no other writers are allowed.
;  472          1460  3  !
;  473          1461  3
;  474          1462  3      IF .CURRENT_WINDOW NEQ 0
;  475          1463  3      THEN
;  476          1464  4          BEGIN
;  477          1465  4          IF NOT .CURRENT_WINDOW [WCB$V_WRITE]
;  478          1466  4              AND NOT .FCB [FCB$V_EXCL]
;  479          1467  4          THEN
;  480          1468  4              IF NOT ARBITRATE_ACCESS (FIB$M_NOWRITE, .FCB)
;  481          1469  4              THEN
;  482          1470  5                  ERR_EXIT (SS$_ACCONFLICT)
;  483          1471  4          END
```

```
;  484          1472  3      ELSE
;  485          1473  3          IF NOT ARBITRATE_ACCESS (FIB$M_NOWRITE, .FCB)
;  486          1474  3          THEN
;  487          1475  3              ERR_EXIT (SS$_ACCONFLICT);
;  488          1476  3
;  489          1477  3      CLEANUP_FLAGS[CLF_REENTER] = 0;             ! from now on deletion proceeds
;  490          1478  3
;  491          1479  3
;  492          1480  3  ! Mark the file for delete. If the file is not accessed, then proceed to
;  493          1481  3  ! actually delete it.
;  494          1482  3  ! In addition, if this is a directory file, clear the directory flag in
;  495          1483  3  ! the header and clean out cached directory data blocks now.
;  496          1484  3  ! Clearing the directory flag in the header allows us to be defensive
;  497          1485  3  ! against accidental directory deletion in delete_file.
;  498          1486  3  !
;  499          1487  3
;  500          1488  3      HEADER[FH2$V_MARKDEL] = 1;
;  501          1489  3
;  502          1490  3      IF TESTBITSC (HEADER [FH2$V_DIRECTORY])
;  503          1491  3      THEN
;  504          1492  3          KILL_BUFFERS (1, .FCB [FCB$L_LOCKBASIS]);
;  505          1493  3
;  506          1494  3      IF MARKDEL_FCB (.FCB)
;  507          1495  3      THEN
;  508          1496  3          DELETE_FILE (.FIB, .HEADER)
;  509          1497  3      ELSE
;  510          1498  4          BEGIN
;  511          1499  4          CHECKSUM (.HEADER);
;  512          1500  4          MARK_DIRTY (.HEADER);
;  513          1501  3          END;
;  514          1502  3
;  515          1503  3  ! The access lock conversion routine is called to:
;  516          1504  3  ! 1) restore the previous lock mode,
;  517          1505  3  ! 2) dequeue the access lock entirely if the refcnt is zero.
;  518          1506  3  ! 3) if the lock was granted exclusive, either restore or dequeue the
;  519          1507  3  !       lock and store the value block.
;  520          1508  3  !
;  521          1509  3
;  522          1510  3      CONV_ACCLOCK (.CURR_LKMODE, .FCB);
;  523          1511  3
;  524          1512  3      IF .FCB [FCB$W_REFCNT] EQL 0
;  525          1513  3      THEN
;  526          1514  4          BEGIN
;  527          1515  4          IF .FCB [FCB$L_DIRINDX] NEQ 0
;  528          1516  4          THEN
;  529          1517  4              KILL_DINDX (.FCB);
;  530          1518  4
;  531          1519  4          DEL_EXTFCB (.FCB);
;  532          1520  4          NUKE_HEAD_FCB (.FCB);
;  533          1521  3          END;
;  534          1522  3
;  535          1523  3      IF .PRIMARY_FCB EQL .FCB THEN PRIMARY_FCB = 0;
;  536          1524  3      IF .DIR_FCB EQL .FCB THEN DIR_FCB = 0;
;  537          1525  3
;  538          1526  3      END         ! of we really do want to delete the file.
;  539          1527  3
;  540          1528  3  ! Otherwise we are just removing a directory entry. If the file name
```

```
;  541          1529  3 ! and back link in the header match the directory, erase the back
;  542          1530  3 ! link.
;  543          1531  3 !
;  544          1532  3
;  545          1533  2 ELSE
;  546          1534  3     BEGIN
;  547          1535  3     CH$MOVE (FID$C_LENGTH, HEADER[FH2$W_BACKLINK], PREV_LINK);
;  548          1536  3     CH$MOVE (FID$C_LENGTH, HEADER[FH2$W_BACKLINK], TEMP_FID);
;  549          1537  3     APPLY_RVN (TEMP_FID[FID$W_RVN], .CURRENT_RVN);
;  550          1538  3     IDENT_AREA = .HEADER + .HEADER[FH2$B_IDOFFSET]*2;
;  551          1539  3     CH$COPY (FI2$S_FILENAME, IDENT_AREA[FI2$T_FILENAME],
;  552          1540  3              ' ', FILENAME_LENGTH+6, PREV_INAME);
;  553          1541  3     IF .HEADER[FH2$B_MPOFFSET] - .HEADER[FH2$B_IDOFFSET]
;  554          1542  3         GEQU ($BYTEOFFSET (FI2$T_FILENAMEXT) + FI2$S_FILENAMEXT) / 2
;  555          1543  3     THEN
;  556          1544  3         CH$MOVE (FI2$S_FILENAMEXT, IDENT_AREA[FI2$T_FILENAMEXT],
;  557          1545  3                                    PREV_INAME[FI2$S_FILENAME]);
;  558          1546  3     IF CH$EQL (FID$C_LENGTH, FIB[FIB$W_DID], FID$C_LENGTH, TEMP_FID)
;  559          1547  3     AND CH$EQL (.RESULT_LENGTH, .RESULT,
;  560          1548  3                 FI2$S_FILENAME+FI2$S_FILENAMEXT, PREV_INAME, ' ')
;  561          1549  3     THEN
;  562          1550  4         BEGIN
;  563          1551  4         HEADER[FH2$W_BK_FIDNUM] = 0;
;  564          1552  4         HEADER[FH2$W_BK_FIDSEQ] = 0;
;  565          1553  4         HEADER[FH2$W_BK_FIDRVN] = 0;
;  566          1554  4         CLEANUP_FLAGS[CLF_FIXLINK] = 1;
;  567          1555  4         CHECKSUM (.HEADER);
;  568          1556  4         MARK_DIRTY (.HEADER);
;  569          1557  3         END;
;  570          1558  2     END;
;  571          1559  2
;  572          1560  2 WRITE_DIRTY (.LB_BASIS [.PRIM_LCKINDX]);
;  573          1561  2
;  574          1562  2 RELEASE_SERIAL_LOCK (.PRIM_LCKINDX);
;  575          1563  2
;  576          1564  2 PRIM_LCKINDX = 0;
;  577          1565  2
;  578          1566  1 END;                                    ! end of routine MARK_DELETE


                                                    .EXTRN  REBLD_PRIM_FCB, BUILD_EXT_FCBS
                                                    .EXTRN  KILL_DINDX, KILL_BUFFERS
                                                    .EXTRN  NUKE_HEAD_FCB, DEL_EXTFCB
                                                    .EXTRN  ARBITRATE_ACCESS
                                                    .EXTRN  CONV_ACCLOCK, WRITE_DIRTY
                                                    .EXTRN  SERIAL_FILE, RELEASE_SERIAL_LOCK
                                                    .EXTRN  SWITCH_VOLUME, SEARCH_FCB
                                                    .EXTRN  CREATE_FCB, READ_HEADER
                                                    .EXTRN  CHECK_PROTECT, WRITE_AUDIT
                                                    .EXTRN  GET_MAP_POINTER
                                                    .EXTRN  READ_BLOCK, INVALIDATE
                                                    .EXTRN  MARK_DIRTY, DELETE_FILE
                                                    .EXTRN  CHECKSUM

                                03FC 00000          .ENTRY  MARK_DELETE, Save R2,R3,R4,R5,R6,R7,R8,R9   ; 1244
                          5E    08   C2 00002       SUBL2   #8, SP                                      ;
```

```
                57      01A8   CA  9E 00005          MOVAB   424(BASE), R7                          ; 1303
                50        04   AC  D0 0000A          MOVL    FIB, R0                               ; 1338
                7E        08   A0  3C 0000E          MOVZWL  8(R0), -(SP)
          0000G   CF           01  FB 00012          CALLS   #1, SWITCH_VOLUME
      7E    04   AC           04  C1 00017          ADDL3   #4, FIB, -(SP)                        ; 1343
          0000G   CF           01  FB 0001C          CALLS   #1, SERIAL_FILE
            18    AA           50  D0 00021          MOVL    R0, 24(BASE)
      7E    04   AC           04  C1 00025          ADDL3   #4, FIB, -(SP)                        ; 1345
          0000G   CF           01  FB 0002A          CALLS   #1, SEARCH_FCB
                53           50  D0 0002F          MOVL    R0, FCB
            C0    AA      80   AA  D0 00032          MOVL    -128(BASE), -64(BASE)                 ; 1346
                6D    0000V   CF  9E 00037          MOVAB   DELETE_HANDLER, (FP)                  ; 1347
                53           DD  0003C              PUSHL   FCB                                   ; 1348
      7E    04   AC           04  C1 0003E          ADDL3   #4, FIB, -(SP)
          0000G   CF           02  FB 00043          CALLS   #2, READ_HEADER
                59           50  D0 00048          MOVL    R0, HEADER
                             6D  D4 0004B          CLRL    (FP)                                  ; 1349
                03        08   AC  E8 0004D          BLBS    DO_DELETE, 1$                         ; 1354
                         0181  31  00051              BRW     21$
                50        04   AC  D0 00054  1$:     MOVL    FIB, R0                               ; 1362
                51        98   AA  D0 00058          MOVL    -104(BASE), R1
                52        4F   A1  9A 0005C          MOVZBL  79(R1), R2
            04  A0              52  B1 00060          CMPW    R2, 4(R0)
                             08  1F 00064          BLSSU   2$
                          09  A0  95 00066          TSTB    9(R0)                                 ; 1363
                             03  12 00069          BNEQ    2$
                             24  BF 0006B          CHMU    #36                                   ; 1364
                             04  00006D            RET
                             52  D4 0006E  2$:     CLRL    FCB_CREATED                           ; 1368
                             53  D5 00070          TSTL    FCB                                   ; 1369
                             0D  12 00072          BNEQ    3$
                52                01  D0 00074          MOVL    #1, FCB_CREATED                   ; 1372
                59                DD 00077          PUSHL   HEADER                            ; 1373
          0000G   CF              01  FB 00079          CALLS   #1, CREATE_FCB
                53              50  D0 0007E          MOVL    R0, FCB
            08  AA              53  D0 00081  3$:     MOVL    FCB, 8(BASE)                      ; 1375
                0D              52  E8 00085          BLBS    FCB_CREATED, 4$                   ; 1381
                10        23    A3  E9 00088          BLBC    35(FCB), 5$                       ; 1385
                        0208    8F  BB 0008C          PUSHR   #^M<R3,R9>                        ; 1389
          0000G   CF              02  FB 00090          CALLS   #2, REBLD_PRIM_FCB
                59              DD 00095  4$:     PUSHL   HEADER                            ; 1391
          0000G   CF              01  FB 00097          CALLS   #1, BUILD_EXT_FCBS
                51        90    AA  D0 0009C  5$:     MOVL    -112(BASE), R1                    ; 1400
                50        04   AC  D0 000A0          MOVL    FIB, R0
      7E    0B  A1           02  00  EF 000A4          EXTZV   #0, #2, 11(R1), -(SP)
                6E        2E   A0  91 000AA          CMPB    46(R0), (SP)
                             04  1B 000AE          BLEQU   6$
                6E        2E   A0  9A 000B0          MOVZBL  46(R0), (SP)
                             53  DD 000B4  6$:     PUSHL   FCB                               ; 1399
                             59  DD 000B6          PUSHL   HEADER
                             02  DD 000B8          PUSHL   #2
          0000G   CF              04  FB 000BA          CALLS   #4, CHECK_PROTECT
        3E    35   A9           05  E1 000BF          BBC     #5, 53(HEADER), 9$                ; 1409
        50    1C   A9           10  9C 000C4          ROTL    #16, 28(HEADER), EOF              ; 1412
                             07  13 000C9          BEQL    7$                                ; 1413
                        20   A9  B5 000CB          TSTW    32(HEADER)                        ; 1414
                             02  12 000CE          BNEQ    7$
```

```
                                50  D7 000D0          DECL    EOF                                             ; 1415
                          01    50  D1 000D2 7$:      CMPL    EOF, #1                                         ; 1416
                                26  1A 000D5          BGTRU   8$                                              ;
                          50    A9  9A 000D7          MOVZBL  1(HEADER), R0                                   ; 1419
                          58  6940  3E 000DB          MOVAW   (HEADER)[R0], MAP_POINTER                       ;
                            0000G 30 000DF          BSBW    GET_MAP_POINTER                                 ; 1420
                                04  DD 000E2          PUSHL   #4                                              ; 1421
                                01  DD 000E4          PUSHL   #1                                              ;
                                57  DD 000E6          PUSHL   LBN                                             ;
                0000G           CF  03  FB 000E8          CALLS   #3, READ_BLOCK                                  ;
                FFFF            8F  60  B1 000ED          CMPW    (BUFFER), #65535                                ; 1422
                                09  12 000F2          BNEQ    8$                                              ;
                                50  DD 000F4          PUSHL   BUFFER                                          ; 1424
                0000G           CF  01  FB 000F6          CALLS   #1, INVALIDATE                                  ;
                                05  11 000FB          BRB     9$                                              ; 1416
                          2174  8F  BF 000FD 8$:      CHMU    #8564                                           ; 1426
                                04  00101          RET                                                     ;
                          52    0924  CA  9E 00102 9$:      MOVAB   2340(BASE), ARGLIST                             ; 1434
                          54    04  D0 00107          MOVL    #4, J                                           ; 1435
                                62  95 0010A 10$:      TSTB    (ARGLIST)                                       ; 1438
                                22  13 0010C          BEQL    11$                                             ;
                          24    A3    02  A2  B1 0010E          CMPW    2(ARGLIST), 36(FCB)                             ; 1439
                                1B  12 00113          BNEQ    11$                                             ;
                          26    A3    04  A2  B1 00115          CMPW    4(ARGLIST), 38(FCB)                             ; 1440
                                14  12 0011A          BNEQ    11$                                             ;
                          28    A3    06  A2  B1 0011C          CMPW    6(ARGLIST), 40(FCB)                             ; 1441
                                0D  12 00121          BNEQ    11$                                             ;
                                52  DD 00123          PUSHL   ARGLIST                                         ; 1444
                0000G           CF  01  FB 00125          CALLS   #1, WRITE_AUDIT                                 ;
                          59    04  AA  D0 0012A          MOVL    4(BASE), HEADER                                 ; 1445
                                06  11 0012E          BRB     12$                                             ; 1446
                          52          10  C0 00130 11$:      ADDL2   #16, ARGLIST                                    ; 1448
                          D4          54  F5 00133          SOBGTR  J, 10$                                          ; 1435
                          52    0B    A3  9A 00136 12$:      MOVZBL  11(FCB), CURR_LKMODE                            ; 1454
                          50    0C    AA  D0 0013A          MOVL    12(BASE), R0                                    ; 1462
                                0A  13 0013E          BEQL    13$                                             ;
                16        0B    A0        01  E0 00140          BBS     #1, 11(R0), 14$                         ; 1465
                11        22    A3        03  E0 00145          BBS     #3, 34(FCB), 14$                        ; 1466
                          51          53  D0 0014A 13$:      MOVL    FCB, R1                                         ; 1473
                          50          01  D0 0014D          MOVL    #1, R0                                          ;
                                0000G 30 00150          BSBW    ARBITRATE_ACCESS                                ;
                          05          50  E8 00153          BLBS    R0, 14$                                         ;
                                0800  8F  BF 00156          CHMU    #2048                                           ; 1475
                                04  0015A          RET                                                     ;
                          02    AA    80  8F  8A 0015B 14$:      BICB2   #128, 2(BASE)                           ; 1477
                          35    A9    80  8F  88 00160          BISB2   #128, 53(HEADER)                        ; 1488
                0A        34    A9        0D  E5 00165          BBCC    #13, 52(HEADER), 15$                    ; 1490
                                4C  A3  DD 0016A          PUSHL   76(FCB)                                         ; 1492
                                01  DD 0016D          PUSHL   #1                                              ;
                          0000G           CF  02  FB 0016F          CALLS   #2, KILL_BUFFERS                                ;
                                53  DD 00174 15$:      PUSHL   FCB                                             ; 1494
                          0000V           CF  01  FB 00176          CALLS   #1, MARKDEL_FCB                                 ;
                                0C          50  E9 0017B          BLBC    R0, 16$                                 ;
                                59  DD 0017E          PUSHL   HEADER                                          ; 1496
                                04  AC  DD 00180          PUSHL   FIB                                             ;
                          0000G           CF  02  FB 00183          CALLS   #2, DELETE_FILE                                 ;
                                0E  11 00188          BRB     17$                                             ;
```

```
                                                59  DD 0018A 16$:    PUSHL   HEADER                                                   ; 1499
                        0000G   CF                      01  FB 0018C        CALLS   #1, CHECKSUM
                                                59  DD 00191        PUSHL   HEADER                                                   ; 1500
                        0000G   CF                      01  FB 00193        CALLS   #1, MARK_DIRTY
                                                0C  BB 00198 17$:    PUSHR   #^M<R2,R3>                                               ; 1510
                        0000G   CF                      02  FB 0019A        CALLS   #2, CONV_ACCLOCK
                                          18    A3  B5 0019F        TSTW    24(FCB)                                                  ; 1512
                                                18  12 001A2        BNEQ    19$
                                        00B0    C3  D5 001A4        TSTL    176(FCB)                                                 ; 1515
                                                07  13 001A8        BEQL    18$
                                                53  DD 001AA        PUSHL   FCB                                                      ; 1517
                        0000G   CF                      01  FB 001AC        CALLS   #1, KILL_DINDX
                                                53  DD 001B1 18$:    PUSHL   FCB                                                      ; 1519
                        0000G   CF                      01  FB 001B3        CALLS   #1, DEL_EXTFCB
                                                53  DD 001B8        PUSHL   FCB                                                      ; 1520
                        0000G   CF                      01  FB 001BA        CALLS   #1, NUKE_HEAD_FCB
                                53      08      AA  D1 001BF 19$:    CMPL    8(BASE), FCB                                             ; 1523
                                                03  12 001C3        BNEQ    20$
                                        08      AA  D4 001C5        CLRL    8(BASE)
                                53      00D0    CA  D1 001C8 20$:    CMPL    208(BASE), FCB                                           ; 1524
                                                7F  12 001CD        BNEQ    25$
                                        00D0    CA  D4 001CF        CLRL    208(BASE)
                                                79  11 001D3        BRB     25$                                                      ; 1354
                30  AA      42  A9              06  28 001D5 21$:    MOVC3   #6, 66(HEADER), 48(BASE)                                 ; 1535
                    6E      42  A9              06  28 001DB        MOVC3   #6, 66(HEADER), TEMP_FID                                 ; 1536
                                        04      AE  95 001E0        TSTB    TEMP_FID+4                                               ; 1537
                                                05  12 001E3        BNEQ    22$
                            04  AE      A0      AA  90 001E5        MOVB    -96(BASE), TEMP_FID+4
                                01      04      AE  91 001EA 22$:    CMPB    TEMP_FID+4, #1
                                                08  12 001EE        BNEQ    23$
                                        A0      AA  D5 001F0        TSTL    -96(BASE)
                                                03  12 001F3        BNEQ    23$
                                        04      AE  94 001F5        CLRB    TEMP_FID+4
                                50              69  9A 001F8 23$:    MOVZBL  (HEADER), R0                                             ; 1538
                                56              6940 3E 001FB        MOVAW   (HEADER)[R0], IDENT_AREA
0056    8F              20      66              14  2C 001FF        MOVC5   #20, (IDENT_AREA), #32, #86, (R7)                        ; 1539
                                                67      00206
                                50      01      A9  9A 00207        MOVZBL  1(HEADER), R0                                            ; 1541
                                51              69  9A 0020B        MOVZBL  (HEADER), R1
                                50              51  C2 0020E        SUBL2   R1, R0
                                3C              50  D1 00211        CMPL    R0, #60                                                  ; 1542
                                                08  1F 00214        BLSSU   24$
                        14  A7      36  A6      0042    8F  28 00216        MOVC3   #66, 54(IDENT_AREA), 20(R7)                      ; 1545
                                50      04      AC  D0 0021E 24$:    MOVL    FIB, R0                                                  ; 1546
                            6E      0A  A0              06  29 00222        CMPC3   #6, 10(R0), TEMP_FID
                                                25  12 00227        BNEQ    25$
0056    8F              20      10  BC      0C      AC  2D 00229        CMPC5   RESULT_LENGTH, @RESULT, #32, #86, (R7)           ; 1547
                                                67      00232
                                                19  12 00233        BNEQ    25$
                                        42      A9  D4 00235        CLRL    66(HEADER)                                               ; 1551
                                        46      A9  B4 00238        CLRW    70(HEADER)                                               ; 1553
                            03  AA      40      8F  88 0023B        BISB2   #64, 3(BASE)                                             ; 1554
                                                59  DD 00240        PUSHL   HEADER                                                   ; 1555
                        0000G   CF                      01  FB 00242        CALLS   #1, CHECKSUM
                                                59  DD 00247        PUSHL   HEADER                                                   ; 1556
                        0000G   CF                      01  FB 00249        CALLS   #1, MARK_DIRTY
                                50      18      AA  D0 0024E 25$:    MOVL    24(BASE), R0                                             ; 1560
```

```
                 0080 CA4C   DD 00252          PUSHL   128(BASE)[R0]                                ;
          0000G  CF           01 FB 00257      CALLS   #1, WRITE_DIRTY                             ;
                    18   AA   DD 0025C         PUSHL   24(BASE)                                    ; 1562
          0000G  CF           01 FB 0025F      CALLS   #1, RELEASE_SERIAL_LOCK                     ;
                    18   AA   D4 00264         CLRL    24(BASE)                                    ; 1564
                              04 00267         RET                                                 ; 1566

; Routine Size:  616 bytes,    Routine Base:  $CODE$ + 006F
```

```
;  20          1567  1 GLOBAL ROUTINE MARKDEL_FCB (FCB) : L_NORM =
; 581          1568  1
; 582          1569  1 !++
; 583          1570  1 !
; 584          1571  1 ! FUNCTIONAL DESCRIPTION:
; 585          1572  1 !
; 586          1573  1 !       This routine marks the FCB for the current file, if any, for delete.
; 587          1574  1 !       In a cluster, it will either mark other FCBs as stale, set the
; 588          1575  1 !       MARKDEL flag in the access lock value block, or both.
; 589          1576  1 !       This routine must be executed in kernel mode.
; 590          1577  1 !
; 591          1578  1 ! CALLING SEQUENCE:
; 592          1579  1 !       MARKDEL_FCB (ARG1)
; 593          1580  1 !
; 594          1581  1 ! INPUT PARAMETERS:
; 595          1582  1 !       ARG1: address of FCB
; 596          1583  1 !
; 597          1584  1 ! IMPLICIT INPUTS:
; 598          1585  1 !       NONE
; 599          1586  1 !
; 600          1587  1 ! OUTPUT PARAMETERS:
; 601          1588  1 !       NONE
; 602          1589  1 !
; 603          1590  1 ! IMPLICIT OUTPUTS:
; 604          1591  1 !       NONE
; 605          1592  1 !
; 606          1593  1 ! ROUTINE VALUE:
; 607          1594  1 !       1 if file may be deleted.
; 608          1595  1 !       0 if delete is to be deferred
; 609          1596  1 !       2 delete is to be deferred and file is accessed on another node
; 610          1597  1 !
; 611          1598  1 ! SIDE EFFECTS:
; 612          1599  1 !       Whether file may be deleted or not, there may be a zero-refcount
; 613          1600  1 !       FCB remaining which must be cleaned up by the caller.
; 614          1601  1 !
; 615          1602  1 !--
; 616          1603  1
; 617          1604  2 BEGIN
; 618          1605  2
; 619          1606  2 MAP
; 620          1607  2         FCB             : REF BBLOCK;   ! FCB arg
; 621          1608  2
; 622          1609  2 BIND_COMMON;
; 623          1610  2
; 624          1611  2 EXTERNAL ROUTINE
; 625          1612  2         LOCK_COUNT      : L_NORM,       ! get count of access locks
; 626          1613  2         QEX_N_CANCEL    : L_NORM;       ! set fcb$v_stale flag in other fcbs.
; 627          1614  2
; 628          1615  2
; 629          1616  2 ! If the FCB exists, we mark it for delete (causing the file to be deleted
; 630          1617  2 ! when the reference count goes to 0).  If the
; 631          1618  2 ! reference count is zero, dump the FCB and its extensions.
; 632          1619  2 !
; 633          1620  2
; 634          1621  2 IF .FCB NEQ 0
; 635          1622  2 THEN
; 636          1623  3     BEGIN
```

```
;  637          1624  3
;  638          1625  3      FCB[FCB$V_MARKDEL] = 1;
;  639          1626  3
;  640          1627  3      IF LOCK_COUNT (.FCB [FCB$L_ACCLKID]) NEQ 1
;  641          1628  3      THEN
;  642          1629  4          BEGIN
;  643          1630  4          IF QEX_N_CANCEL (.FCB [FCB$L_ACCLKID])
;  644          1631  4
;  645          1632  4  ! Normally the lock will not actually be granted from the qex_n_cancel call.
;  646          1633  4  ! If it is granted though (success), then set the lockmode field in the
;  647          1634  4  ! fcb so that the subsequent conv_acclock handles the value block correctly.
;  648          1635  4  !
;  649          1636  4
;  650          1637  4          THEN
;  651          1638  4              FCB [FCB$B_ACCLKMODE] = LCK$K_EXMODE;
;  652          1639  4
;  653          1640  4          RETURN 2
;  654          1641  3          END;
;  655          1642  3
;  656          1643  3      IF .FCB[FCB$W_REFCNT] NEQ 0
;  657          1644  3      THEN
;  658          1645  3          RETURN 0;                ! file still accessed here
;  659          1646  3
;  660          1647  2      END;
;  661          1648  2
;  662          1649  2  RETURN 1;                                ! ok to delete file
;  663          1650  2
;  664          1651  1  END;                                     ! end of routine MARKDEL_FCB


                                                .EXTRN  LOCK_COUNT, QEX_N_CANCEL

                                  0000 00000    .ENTRY  MARKDEL_FCB, Save nothing                ; 1567
        50      04    AC  D0 00002            MOVL    FCB, R0                                  ; 1621
                      39  13 00006            BEQL    3$
     22 A0            02  88 00008            BISB2   #2, 34(R0)                               ; 1625
        50      04    AC  D0 0000C            MOVL    FCB, R0                                  ; 1627
                      48  A0 DD 00010         PUSHL   72(R0)
  0000G CF            01  FB 00013            CALLS   #1, LOCK_COUNT
        01            50  D1 00018            CMPL    R0, #1
                      1B  13 0001B            BEQL    2$
        50      04    AC  D0 0001D            MOVL    FCB, R0                                  ; 1630
                      48  A0 DD 00021         PUSHL   72(R0)
  0000G CF            01  FB 00024            CALLS   #1, QEX_N_CANCEL
        08            50  E9 00029            BLBC    R0, 1$
        50      04    AC  D0 0002C            MOVL    FCB, R0                                  ; 1638
     0B A0            05  90 00030            MOVB    #5, 11(R0)
        50            02  D0 00034 1$:        MOVL    #2, R0                                   ; 1640
                          04 00037            RET
        50      04    AC  D0 00038 2$:        MOVL    FCB, R0                                  ; 1643
                18    A0  B5 0003C            TSTW    24(R0)
                      04  12 0003F            BNEQ    4$
        50            01  D0 00041 3$:        MOVL    #1, R0                                   ; 1649
                          04 00044            RET
                      50  D4 00045 4$:        CLRL    R0                                       ; 1651
                          04 00047            RET
```

```
; Routine Size:  72 bytes,    Routine Base:  $CODE$ + 02D7
```

```
;  666     1652  1 ROUTINE DELETE_HANDLER (SIGNAL, MECHANISM) : L_NORM =
;  667     1653  1
;  668     1654  1 !++
;  669     1655  1 !
;  670     1656  1 ! FUNCTIONAL DESCRIPTION:
;  671     1657  1 !
;  672     1658  1 !       This routine is the condition handler for reading the file header.
;  673     1659  1 !       If any errors occurr, it unwinds and returns to MARK_DELETE's caller,
;  674     1660  1 !       causing the delete of a bad file header to be a quiet NOP.
;  675     1661  1 !
;  676     1662  1 !
;  677     1663  1 ! CALLING SEQUENCE:
;  678     1664  1 !       HANDLER (ARG1, ARG2)
;  679     1665  1 !
;  680     1666  1 ! INPUT PARAMETERS:
;  681     1667  1 !       ARG1: address of signal array
;  682     1668  1 !       ARG2: address of mechanism array
;  683     1669  1 !
;  684     1670  1 ! IMPLICIT INPUTS:
;  685     1671  1 !       NONE
;  686     1672  1 !
;  687     1673  1 ! OUTPUT PARAMETERS:
;  688     1674  1 !       NONE
;  689     1675  1 !
;  690     1676  1 ! IMPLICIT OUTPUTS:
;  691     1677  1 !       NONE
;  692     1678  1 !
;  693     1679  1 ! ROUTINE VALUE:
;  694     1680  1 !       SS$_RESIGNAL or none if unwind
;  695     1681  1 !
;  696     1682  1 ! SIDE EFFECTS:
;  697     1683  1 !       NONE
;  698     1684  1 !
;  699     1685  1 !--
;  700     1686  1
;  701     1687  1
;  702     1688  2 BEGIN
;  703     1689  2
;  704     1690  2 MAP
;  705     1691  2         SIGNAL          : REF BBLOCK,   ! signal arg array
;  706     1692  2         MECHANISM       : REF BBLOCK;   ! mechanism arg array
;  707     1693  2
;  708     1694  2 BIND_COMMON;
;  709     1695  2
;  710     1696  2 ! If the condition is change mode to user (error exit) cause an unwind to
;  711     1697  2 ! return to DELETE's caller.
;  712     1698  2 ! Otherwise, just resignal the condition.
;  713     1699  2 !
;  714     1700  2
;  715     1701  2 IF .SIGNAL[CHF%L_SIG_NAME] EQL SS$_CMODUSER
;  716     1702  2 THEN
;  717     1703  3     BEGIN
;  718     1704  3     USER_STATUS = . SAVE_STATUS;
;  719     1705  3     $UNWIND ();
;  720     1706  2     END;
;  721     1707  2
;  722     1708  2 RETURN SS$_RESIGNAL;                    ! status is irrelevant if unwinding
```

```
;  723          1709  2
;  724          1710  1 END;                                    ! end of routine DELETE_HANDLER


                                                                .EXTRN  SYS$UNWIND

                                                0000 00000 DELETE_HANDLER:
                                                                .WORD   Save nothing                    ; 1652
                                    50      04  AC  D0 00002           MOVL    SIGNAL, R0              ; 1701
                          00000424  8F      04  A0  D1 00006           CMPL    4(R0), #1060            ;
                                                0E  12 0000E           BNEQ    1$                      ;
                               80   AA      C0  AA  D0 00010           MOVL    -64(BASE), -128(BASE)   ; 1704
                                                7E  7C 00015           CLRQ    -(SP)                   ; 1705
                          00000000G 00          02  FB 00017           CALLS   #2, SYS$UNWIND          ;
                                    50    0918  8F  3C 0001E 1$:       MOVZWL  #2328, R0               ; 1708
                                                04  00023           RET                             ; 1710

; Routine Size:  36 bytes,    Routine Base:  $CODE$ + 031F

;  725          1711  1
;  726          1712  1 END
;  727          1713  0 ELUDOM




;                                       PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;  $CODE$                         835  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                                   Library Statistics
;
;                                           -------- Symbols --------      Pages       Processing
;       File                                Total    Loaded   Percent      Mapped      Time
;  _$255$DUA28:[SYSLIB]LIB.L32;1            18619        74         0      1000          00:01.9




;                                       COMMAND QUALIFIERS
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:DELETE/OBJ=OBJ$:DELETE MSRC$:DELETE/UPDATE=(ENH$:DELETE)
; Size:         835 code + 0 data bytes
```

```
; Run Time:          00:50.6
; Elapsed Time:      01:48.1
; Lines/CPU Min:      2033
; Lexemes/CPU-Min:  56607
; Memory Used:   352 pages
; Compilation Complete
```

DEACCS
LIS

DELETE
LIS

DIRSCN
LIS

DISPAT
LIS

CREHDR
LIS

DIRACC
LIS

DELBAD
LIS

CREFCB
LIS

CREWIN
LIS

DISPATCH
LIS

ENTER
LIS

DELFIL
LIS