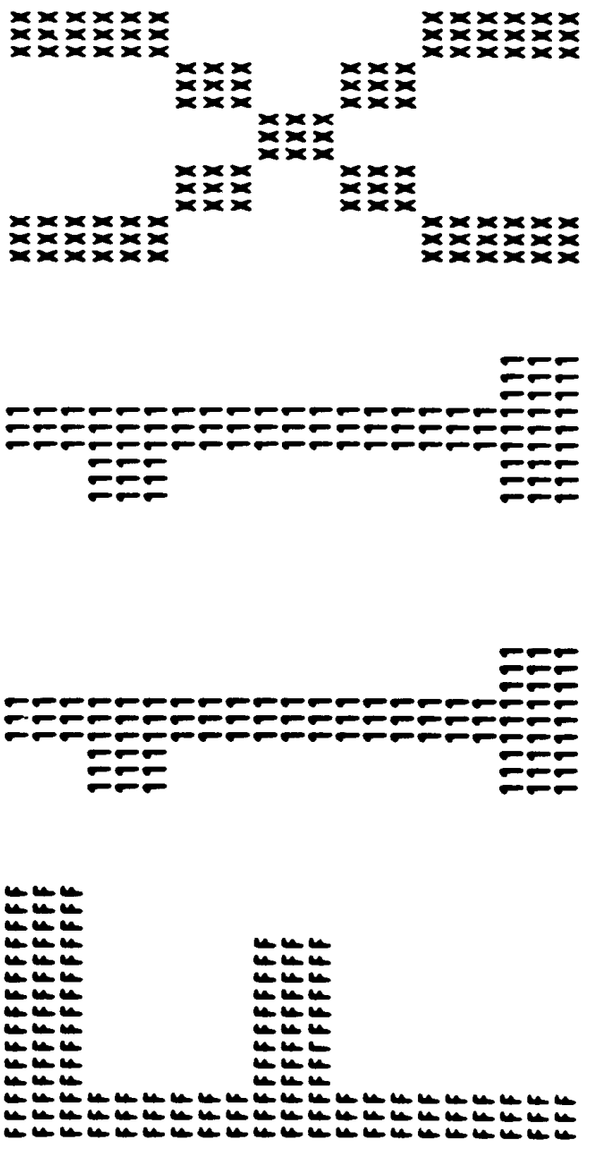

FILEID**TRUNC

```
TTTTTTTTTT  RRRRRRRR    UU      UU  NN      NN    CCCCCCCC
TTTTTTTTTT  RRRRRRRR    UU      UU  NN      NN    CCCCCCCC
    TT      RR      RR  UU      UU  NN      NN  CC
    TT      RR      RR  UU      UU  NN      NN  CC
    TT      RR      RR  UU      UU  NNNN    NN  CC
    TT      RR      RR  UU      UU  NNNN    NN  CC
    TT      RRRRRRRR    UU      UU  NN  NN  NN  CC
    TT      RRRRRRRR    UU      UU  NN  NN  NN  CC
    TT      RR  RR      UU      UU  NN    NNNN  CC
    TT      RR  RR      UU      UU  NN    NNNN  CC
    TT      RR    RR    UU      UU  NN      NN  CC              ....
    TT      RR    RR    UU      UU  NN      NN  CC              ....
    TT      RR      RR  UUUUUUUUUU  NN      NN    CCCCCCCC    ....
    TT      RR      RR  UUUUUUUUUU  NN      NN    CCCCCCCC    ....

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

```
;     1          0001  0 MODULE TRUNC (
;     2          0002  0                 LANGUAGE (BLISS32),
;001 !CDS0008    0003  0                 IDENT = 'V04-001'
;    4-1         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 truncates a file by deallocating the indicated blocks.
;    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:  21-Mar-1977  10:41
;    48          0048  1 !
;    49          0049  1 ! MODIFIED BY:
;    50          0050  1 !
;001 !CDS0008    0051  1 !       V04-001 CDS0008         Christian D. Saether     6-Nov-1984
;002 !CDS0008    0052  1 !               Fix observed bug which causes PRIMARY_FCB to get
;003 !CDS0008    0053  1 !               the wrong header lbn stored into it after deallocating
;004 !CDS0008    0054  1 !               blocks to the bad block file.  This also fixes
;005 !CDS0008    0055  1 !               problems which would occur if either the bad block
;006 !CDS0008    0056  1 !               or bad log files get into multiple headers and we
;007 !CDS0008    0057  1 !               are running in a minimal buffer environment.
```

```
:008 !CDS0008    0058  1 !
:    51          0059  1 !       V03-013 CDS0007         Christian D. Saether    14-Aug-1984
:    52          0060  1 !               Remove reference to update_filesize routine.
:    53          0061  1 !
:    54          0062  1 !       V03-012 CDS0006         Christian D. Saether    31-July-1984
:    55          0063  1 !               Remove local declaration of get_map_pointer linkage.
:    56          0064  1 !
:    57          0065  1 !       V03-011 CDS0005         Christian D. Saether     5-July-1984
:    58          0066  1 !               Do not call READ_HEADER with the file id argument
:    59          0067  1 !               when re-reading primary header at the end because
:    60          0068  1 !               we always have a primary_fcb now and when this
:    61          0069  1 !               routine is called from deaccess on a deferred
:    62          0070  1 !               truncate, the fid field is not filled in.
:    63          0071  1 !
:    64          0072  1 !       V03-010 CDS0004         Christian D. Saether    22-Apr-1984
:    65          0073  1 !               Change linkage L_TRUNC_CHECKS to L_JSB_2ARGS.
:    66          0074  1 !
:    67          0075  1 !       V03-009 CDS0003         Christian D. Saether    30-Dec-1983
:    68          0076  1 !               Use L_NORM linkage and BIND_COMMON macro.
:    69          0077  1 !
:    70          0078  1 !       V03-008 CDS0002         Christian D. Saether    25-Sep-1983
:    71          0079  1 !               Manually merge in code associated with STJ3097.
:    72          0080  1 !
:    73          0081  1 !       V03-007 STJ3097         Steven T. Jeffreys,     29-Apr-1983
:    74          0082  1 !               Refinement of STJ3072.  Only do the erase if the
:    75          0083  1 !               volume or file ERASE attribute is set.
:    76          0084  1 !
:    77          0085  1 !       V03-006 CDS0001         Christian D. Saether    21-Apr-1983
:    78          0086  1 !               Break out initial error checks into separate routine.
:    79          0087  1 !               Make the truncation vbn an input argument.
:    80          0088  1 !
:    81          0089  1 !       V03-005 STJ3072         Steven T. Jeffreys,     25-Mar-1983
:    82          0090  1 !               Erase blocks returned to the storage map.  Later this
:    83          0091  1 !               will be conditionalized.
:    84          0092  1 !
:    85          0093  1 !       V03-004 ACG0299         Andrew C. Goldstein,    12-Oct-1982  17:21
:    86          0094  1 !               Make truncate tolerant of bad map pointer use count
:    87          0095  1 !
:    88          0096  1 !       V03-003 ACG0296         Andrew C. Goldstein,    8-Jul-1982  21:32
:    89          0097  1 !               Fix truncation of placed allocation pointers
:    90          0098  1 !
:    91          0099  1 !       V03-002 ACG0287         Andrew C. Goldstein,    14-Apr-1982  17:16
:    92          0100  1 !               Check for index file in header rather than FCB
:    93          0101  1 !
:    94          0102  1 !       V03-001 LMP0023         L. Mark Pilant,          7-Apr-1982  16:45
:    95          0103  1 !               Give a privilege violation if attempting to truncate the
:    96          0104  1 !               index file (INDEXF.SYS).
:    97          0105  1 !
:    98          0106  1 !       V02-011 ACG35898        Andrew C. Goldstein,    10-Mar-1981  22:15
:    99          0107  1 !               Update HIBLK in the primary header
:   100          0108  1 !
:   101          0109  1 !       V02-010 ACG0170         Andrew C. Goldstein,    7-May-1980  18:27
:   102          0110  1 !               Fix handling of map pointer use count
:   103          0111  1 !
:   104          0112  1 !       V02-009 ACG0167         Andrew C. Goldstein,    16-Apr-1980  19:28
:   105          0113  1 !               Previous revision history moved to F11B.REV
:   106          0114  1 !**
```

```
;  107          0115  1
;  108          0116  1
;  109          0117  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;  110          0118  1 REQUIRE 'SRC$:FCPDEF.B32';
;  111          1109  1
;  112          1110  1
;  113          1111  1 FORWARD ROUTINE
;  114          1112  1         TRUNCATE        : L_NORM NOVALUE, ! truncate file
;  115          1113  1         TRUNCATE_HEADER : L_NORM NOVALUE, ! truncate individual file header
;  116          1114  1         TRUNC_CHECKS    : L_JSB_2ARGS NOVALUE ; ! initial truncation error checks.
```

```
;  118          1115  1 GLOBAL ROUTINE TRUNCATE (FIB, FILEHEADER, TRNVBN) : L_NORM NOVALUE =
;  119          1116  1
;  120          1117  1 !++
;  121          1118  1 !
;  122          1119  1 ! FUNCTIONAL DESCRIPTION:
;  123          1120  1 !
;  124          1121  1 !       This routine truncates a file to the size indicated in the FIB by
;  125          1122  1 !       deallocating the necessary blocks. The erase flag controls whether
;  126          1123  1 !       the retrieval pointers are erased in the header. The deallocate flag
;  127          1124  1 !       controls whether or not the blocks are actually returned to the
;  128          1125  1 !       storage map.
;  129          1126  1 !
;  130          1127  1 ! CALLING SEQUENCE:
;  131          1128  1 !       TRUNCATE (ARG1, ARG2, ARG3)
;  132          1129  1 !
;  133          1130  1 ! INPUT PARAMETERS:
;  134          1131  1 !       ARG1: address of FIB for operation
;  135          1132  1 !       ARG2: address of file header
;  136          1133  1 !       ARG3: VBN to truncate from
;  137          1134  1 !
;  138          1135  1 ! IMPLICIT INPUTS:
;  139          1136  1 !       CURRENT_VCB: VCB of volume
;  140          1137  1 !
;  141          1138  1 ! OUTPUT PARAMETERS:
;  142          1139  1 !       NONE
;  143          1140  1 !
;  144          1141  1 ! IMPLICIT OUTPUTS:
;  145          1142  1 !       NONE
;  146          1143  1 !
;  147          1144  1 ! ROUTINE VALUE:
;  148          1145  1 !       NONE
;  149          1146  1 !
;  150          1147  1 ! SIDE EFFECTS:
;  151          1148  1 !       storage bitmap altered
;  152          1149  1 !       file header altered
;  153          1150  1 !
;  154          1151  1 !--
;  155          1152  1
;  156          1153  2 BEGIN
;  157          1154  2
;  158          1155  2 MAP
;  159          1156  2         FIB             : REF BBLOCK,   ! FIB for operation
;  160          1157  2         FILEHEADER      : REF BBLOCK;   ! file header
;  161          1158  2
;  162          1159  2 LINKAGE
;  163          1160  2         L_MAKE_POINTER  = CALL :
;  164          1161  2                           GLOBAL (PREV_POINTER = 9);
;  165          1162  2
;  166          1163  2 GLOBAL REGISTER
;  167          1164  2         COUNT           = 6,            ! count of blocks returned
;  168          1165  2         LBN             = 7,            ! LBN of map entry
;  169          1166  2         MAP_POINTER     = 8 : REF BBLOCK, ! pointer to scan map
;  170          1167  2         PREV_POINTER    = 9 : REF BBLOCK; ! pointer to build new map entry
;  171          1168  2
;  172          1169  2 LOCAL
;  173          1170  2         FCB             : REF BBLOCK,   ! FCB of current file header
;  174          1171  2         HEADER          : REF BBLOCK,   ! address of current file header
```

```
;  175          1172  2          ALT_HEADER      : REF BBLOCK,    ! address of header copy to free blocks
;  176          1173  2          NEW_HEADER      : REF BBLOCK,    ! address of extension file header
;  177          1174  2          TRUNC_POINTER,                   ! pointer to start of truncation
;  178          1175  2          MAP_END,                         ! pointer to end of map area
;  179          1176  2          VBN,                             ! relative VBN of operation
;  180          1177  2          HEADER_VBN,                      ! value of VBN at start of this header
;  181          1178  2          HEADER_SIZE,                     ! number of blocks mapped by header
;  182          1179  2          EXT_FID         : BBLOCK [FID$C_LENGTH], ! file ID of extension header
;  183          1180  2          EX_SEGNUM,                       ! segment number of ext header
;  184          1181  2          REREAD,                          ! flag to reread primary header
;  185          1182  2          REREAD2;                         ! flag to update primary header
;  186          1183  2
;  187          1184  2 LABEL
;  188          1185  2          DO_TRUNCATE,                     ! main body of truncate processing code
;  189          1186  2          VBN_LOOP;                        ! main loop to scan for starting VBN
;  190          1187  2
;  191          1188  2 BIND_COMMON;
;  192          1189  2
;  193          1190  2 EXTERNAL ROUTINE
;  194          1191  2          PMS_START_SUB   : L_NORM,        ! start subfunction metering
;  195          1192  2          PMS_END_SUB     : L_NORM,        ! end subfunction metering
;  196          1193  2          FILE_SIZE       : L_NORM,        ! compute size mapped by header
;  197          1194  2          SEARCH_FCB      : L_NORM,        ! search FCB list for FCB
;  198          1195  2          CHARGE_QUOTA    : L_NORM,        ! charge blocks to user's quota
;  199          1196  2          DEALLOCATE_BAD  : L_NORM ADDRESSING_MODE (GENERAL),
;  200          1197  2                                           ! mark blocks bad
;  201          1198  2          MARK_DIRTY      : L_NORM,        ! mark buffer for write-back
;  202          1199  2          CHECKSUM        : L_NORM,        ! checksum file header
;  203          1200  2          GET_MAP_POINTER : L_MAP_POINTER, ! get value of next map pointer
;  204          1201  2          MAKE_POINTER    : L_MAKE_POINTER, ! build a new map pointer
;  205          1202  2          NEXT_HEADER     : L_NORM,        ! read next extension header
;  206          1203  2          CREATE_BLOCK    : L_NORM,        ! allocate a block buffer
;  207          1204  2          INVALIDATE      : L_NORM,        ! invalidate a block buffer
;  208          1205  2          INIT_FCB2       : L_NORM,        ! initialize FCB
;  209          1206  2          WRITE_HEADER    : L_NORM,        ! write file header
;  210          1207  2          READ_HEADER     : L_NORM,        ! read file header
;  211          1208  2          DEL_EXTFCB      : L_NORM,        ! delete extension FCB's
;  212          1209  2          DELETE_FILE     : L_NORM;        ! delete remainder of file
;  213          1210  2
;  214          1211  2
;  215          1212  2 ! Start metering for this subfunction.
;  216          1213  2 !
;  217          1214  2
;  218          1215  2 PMS_START_SUB (PMS_ALLOC);
;  219          1216  2
;  220          1217  2 TRUNC_CHECKS (.FIB, .FILEHEADER);
;  221          1218  2
;  222          1219  2   Establish the basic pointers. Round up the starting VBN to the next cluster
;  223          1220  2 ! boundary and adjust it to a zero start.
;  224          1221  2 ! Round down the file size.
;  225          1222  2 !
;  226          1223  2
;  227          1224  2 HEADER = .FILEHEADER;
;  228          1225  2 FCB = .PRIMARY_FCB;
;  229          1226  2 VBN = .TRNVBN - 1;
;  230          1227  2
;  231          1228  2 ! Init the user's return parameters.
```

```
;  232          1229  2 !
;  233          1230  2
;  234          1231  2 FIB[FIB$L_EXVBN] = 1;
;  235          1232  2
;  236          1233  2 REREAD = 0;
;  237          1234  2
;  238          1235  2 ! Now scan the file headers for the retrieval pointer containing the starting
;  239          1236  2 ! VBN. If the VBN is off the end of file, report the error; if it coincides,
;  240          1237  2 ! the operation is a noop.
;  241          1238  2 !
;  242          1239  2
;  243          1240  2 DO_TRUNCATE:
;  244          1241  3 BEGIN
;  245          1242  3
;  246          1243  3 VBN_LOOP:
;  247          1244  4 BEGIN
;  248          1245  4 WHILE 1 DO
;  249          1246  5     BEGIN
;  250          1247  5     MAP_POINTER = .HEADER + .HEADER[FH2$B_MPOFFSET]*2;
;  251          1248  5     MAP_END = .MAP_POINTER + .HEADER[FH2$B_MAP_INUSE]*2;
;  252          1249  5     PREV_POINTER = .MAP_POINTER;
;  253          1250  5     HEADER_VBN = .VBN;
;  254          1251  5
;  255          1252  5     UNTIL .MAP_POINTER GEQA .MAP_END DO
;  256          1253  6         BEGIN
;  257          1254  6         GET_MAP_POINTER ();
;  258          1255  6         IF .COUNT GEQU .VBN THEN LEAVE VBN_LOOP;
;  259          1256  6         VBN = .VBN - .COUNT;
;  260          1257  6         FIB[FIB$L_EXVBN] = .FIB[FIB$L_EXVBN] + .COUNT;
;  261          1258  6         IF .COUNT NEQ 0
;  262          1259  6         THEN PREV_POINTER = .MAP_POINTER;
;  263          1260  5         END;
;  264          1261  5
;  265          1262  5 ! We have scanned through an entire header. Chain to the next header if it
;  266          1263  5 ! exists. If we run out of headers, then the truncate point is beyond end
;  267          1264  5 ! of file.
;  268          1265  5 !
;  269          1266  5
;  270          1267  5     NEW_HEADER = NEXT_HEADER (.HEADER, .FCB);
;  271          1268  5     IF .NEW_HEADER EQL 0 THEN EXITLOOP;
;  272          1269  5     REREAD = 1;
;  273          1270  5     HEADER = .NEW_HEADER;
;  274          1271  5
;  275          1272  5     IF .FCB NEQ 0
;  276          1273  5     THEN FCB = .FCB[FCB$L_EXFCB];
;  277          1274  4     END;                                    ! end of header scan loop
;  278          1275  4
;  279          1276  4 IF .VBN NEQ 0
;  280          1277  5 THEN ERR_EXIT (SS$_ENDOFFILE)
;  281          1278  3 END;                                        ! end of VBN_LOOP
;  282          1279  3
;  283          1280  3 ! We are now pointing at the retrieval pointer in which the truncation starts.
;  284          1281  3 ! VBN contains the number of blocks to retain in that pointer. We must now
;  285          1282  3 ! round it down or up to the next cluster boundary, depending on whether or
;  286          1283  3 ! not the blocks are to be marked bad, respectively.
;  287          1284  3 !
;  288          1285  3
```

```
;  289          1286  3 USER_STATUS[1] = - .VBN;
;  290          1287  5 VBN = ((.VBN
;  291          1288  6          + (IF NOT .FIB[FIB$V_MARKBAD]
;  292          1289  6             THEN .CURRENT_VCB[VCB$W_CLUSTER] - 1
;  293          1290  6             ELSE 0)
;  294          1291  5          )
;  295          1292  3          / .CURRENT_VCB[VCB$W_CLUSTER]) * .CURRENT_VCB[VCB$W_CLUSTER];
;  296          1293  3 USER_STATUS[1] = .USER_STATUS[1] + .VBN;
;  297          1294  3 FIB[FIB$L_EXVBN] = .FIB[FIB$L_EXVBN] + .VBN;
;  298          1295  3 HEADER_VBN = .HEADER_VBN + .USER_STATUS[1];
;  299          1296  3
;  300          1297  3 ! See if rounding up is causing us to keep the entire map pointer. If so,
;  301          1298  3 ! bump to the next pointer. If that takes us to the end of the map area of
;  302          1299  3 ! a header with no extension, return with no action. (This case is common
;  303          1300  3 ! enough to be worth checking for.)
;  304          1301  3 !
;  305          1302  3
;  306          1303  3 IF .VBN EQL .COUNT
;  307          1304  3 THEN
;  308          1305  4     BEGIN
;  309          1306  4     VBN = 0;
;  310          1307  4     PREV_POINTER = .MAP_POINTER;
;  311          1308  4     IF .PREV_POINTER GEQA .MAP_END
;  312          1309  4     AND .HEADER[FH2$W_EX_FIDNUM] EQL 0
;  313          1310  4     AND .HEADER[FH2$W_EX_FIDRVN] EQL 0
;  314          1311  4     THEN LEAVE DO_TRUNCATE;
;  315          1312  3     END;
;  316          1313  3
;  317          1314  3 ! If we are turning blocks over to the bad block file, check that
;  318          1315  3 ! (1) the pointer given is the last pointer in the header.
;  319          1316  3 ! (2) the header is the last one for the file, and (3) that the quantity
;  320          1317  3 ! being deallocated is exactly one cluster, this being the only condition
;  321          1318  3 ! we can correctly handle.
;  322          1319  3 !
;  323          1320  3
;  324          1321  3 IF .FIB[FIB$V_MARKBAD]
;  325          1322  3 THEN
;  326          1323  3     IF .MAP_POINTER NEQ .MAP_END
;  327          1324  3     OR .COUNT - .VBN NEQ .CURRENT_VCB[VCB$W_CLUSTER]
;  328          1325  3     OR .HEADER[FH2$W_EX_FIDNUM] NEQ 0
;  329          1326  3     OR .HEADER[FH2$W_EX_FIDSEQ] NEQ 0
;  330          1327  3     THEN ERR_EXIT (SS$_BADPARAM);
;  331          1328  3
;  332          1329  3 ! Do the real truncate. Set up cleanup status and get control blocks in shape.
;  333          1330  3 !
;  334          1331  3
;  335          1332  3 CLEANUP_FLAGS[CLF_FIXFCB] = 1;
;  336          1333  3 CLEANUP_FLAGS[CLF_INVWINDOW] = 1;
;  337          1334  3 PRIMARY_FCB [FCB$L_FILESIZE] = .FIB[FIB$L_EXVBN] - 1;
;  338          1335  3
;  339          1336  3 ! Update the HIBLK field in the record attributes to reflect the new file
;  340          1337  3 ! size, if this is the primary header..
;  341          1338  3 !
;  342          1339  3
;  343          1340  3 IF NOT .REREAD
;  344          1341  3 THEN BBLOCK [HEADER[FH2$W_RECATTR], FAT$L_HIBLK] = ROT (.FIB[FIB$L_EXVBN]-1, 16);
;  345          1342  3
```

```
;  346          1343  3 ! Make a copy of the file header with which to free the blocks. In the original,
;  347          1344  3 ! zero out the map pointers being freed and write the header back before
;  348          1345  3 ! deallocating the blocks, so that we do not get a file header mapping free
;  349          1346  3 ! blocks if the system crashes while this is going on.
;  350          1347  3 !
;  351          1348  3
;  352          1349  3 ALT_HEADER = CREATE_BLOCK (-1, 1, HEADER_TYPE);
;  353          1350  3 INVALIDATE (.ALT_HEADER);
;  354          1351  3 CH$MOVE (512, .HEADER, .ALT_HEADER);
;  355          1352  3 TRUNC_POINTER = .PREV_POINTER - .HEADER + .ALT_HEADER;
;  356          1353  3
;  357          1354  3 HEADER[FH2$B_MAP_INUSE] = (.PREV_POINTER - .HEADER) / 2 - .HEADER[FH2$B_MPOFFSET];
;  358          1355  3 IF .VBN NEQ 0
;  359          1356  3 THEN
;  360          1357  4     BEGIN
;  361          1358  4     MAP_POINTER = .PREV_POINTER;
;  362          1359  4     GET_MAP_POINTER ();
;  363          1360  4     MAKE_POINTER (.VBN, .LBN, .HEADER,
;  364          1361  5                   (IF .PREV_POINTER[FM2$V_FORMAT] EQL FM2$C_PLACEMENT
;  365          1362  5                    THEN .PREV_POINTER[FM2$W_WORD0]
;  366          1363  4                    ELSE 0));
;  367          1364      END;
;  368          1365  3 MAP_END = .HEADER + .HEADER[FH2$B_ACOFFSET]*2;
;  369          1366  3 IF .MAP_END - .PREV_POINTER GTR 0
;  370          1367  3 THEN CH$FILL (0, .MAP_END - .PREV_POINTER, .PREV_POINTER);
;  371          1368  3
;  372          1369  3 EX_SEGNUM = .HEADER[FH2$W_SEG_NUM] + 1;
;  373          1370  3 CH$MOVE (FID$C_LENGTH, HEADER[FH2$W_EXT_FID], EXT_FID);
;  374          1371  3 CH$FILL (0, FID$C_LENGTH, HEADER[FH2$W_EXT_FID]);
;  375          1372  3 CHECKSUM (.HEADER);
;  376          1373  3 WRITE_HEADER ();
;  377          1374  3
;  378          1375  3 IF .FCB NEQ 0 AND .FCB NEQ .PRIMARY_FCB
;  379          1376  3 THEN KERNEL_CALL (INIT_FCB2, .FCB, .HEADER);
;  380          1377  3
;  381          1378  3 ! Compute the number of blocks being deallocated and credit them to the
;  382          1379  3 ! file owner. We compute this by taking the total space mapped by the header,
;  383          1380  3 ! less the number of blocks passed over in the scan.
;  384          1381  3 !
;  385          1382  3
;  386          1383  3 IF NOT .CLEANUP_FLAGS[CLF_NOTCHARGED]
;  387          1384  3 THEN
;  388          1385  4     BEGIN
;  389          1386  4     HEADER_SIZE = FILE_SIZE (.ALT_HEADER);
;  390          1387  4     CHARGE_QUOTA (.HEADER[FH2$L_FILEOWNER], - (.HEADER_SIZE - .HEADER_VBN),
;  391          1388  4                   BITLIST (QUOTA_CHARGE));
;  392          1389  3     END;
;  393          1390  3
;001 !CDS0008   1391  3 REREAD2 = .REREAD;
;002 !CDS0008   1392  3
;  394          1393  3 ! Now we can free the blocks being truncated off. They are turned over
;  395          1394  3 ! either to the storage map, or to the bad block file.
;  396          1395  3 !
;  397          1396  3
;  398          1397  3 IF .FIB[FIB$V_MARKBAD]
;  399          1398  3 THEN
;001 !CDS0008   1399  4     BEGIN
```

```
:002 !CDS0008    1400  4     DEALLOCATE_BAD (.FIB, .ALT_HEADER, .TRUNC_POINTER, .VBN);
:003 !CDS0008    1401  4
:004 !CDS0008    1402  4 ! We need to re-read the primary header because the headers for the
:005 !CDS0008    1403  4 ! badblk.sys and badlog.sys files were most likely read in deallocate_bad
:006 !CDS0008    1404  4 ! and they have set the cell HEADER_LBN with the wrong lbn, which will
:007 !CDS0008    1405  4 ! get stored (incorrectly) in the primary_fcb when the update_fcb routine
:008 !CDS0008    1406  4 ! is called at the end of the MODIFY routine.
:009 !CDS0008    1407  4 ! The re-read also needs to be done in case either badblk or badlog have
:010 !CDS0008    1408  4 ! multiple headers.
:011 !CDS0008    1409  4 !
:012 !CDS0008    1410  4
:013 !CDS0008    1411  4     REREAD = 1;
:014 !CDS0008    1412  4     END
:015 !CDS0008    1413  3 ELSE
:016 !CDS0008    1414  3     TRUNCATE_HEADER (.FIB, .ALT_HEADER, .TRUNC_POINTER, .VBN);
:017 !CDS0008    1415  3
:   405-5        1416  3 IF .EXT_FID[FID$W_NUM] NEQ 0
:   406          1417  3 OR .EXT_FID[FID$W_RVN] NEQ 0
:   407          1418  3 THEN
:   408          1419  4     BEGIN
:   409          1420  4     REREAD = 1;
:   410          1421  4     HEADER = NEXT_HEADER (0, .FCB, EXT_FID, .EX_SEGNUM);
:   411          1422  4     KERNEL_CALL (DEL_EXTFCB, .FCB);
:   412          1423  4     DELETE_FILE (.FIB, .HEADER);
:   413          1424  3     END;
:   414          1425  3
:   415          1426  2 END;                                    ! end of block DO_TRUNCATE
:   416          1427  2
:   417          1428  2 ! If this was a truncate of a multi-header file, reread the primary header
:   418          1429  2 ! and update the HIBLK field in the record attributes to reflect the new file
:   419          1430  2 ! size.
:   420          1431  2 !
:   421          1432  2
:   422          1433  2 IF .REREAD
:   423          1434  2 THEN HEADER = READ_HEADER (0, .PRIMARY_FCB);
:   424          1435  2
:   425          1436  2 IF .REREAD2
:   426          1437  2 THEN
:   427          1438  3     BEGIN
:   428          1439  3     BBLOCK [HEADER[FH2$W_RECATTR], FAT$L_HIBLK] = ROT (.FIB[FIB$L_EXVBN]-1, 16);
:   429          1440  3     MARK_DIRTY (.HEADER);
:   430          1441  2     END;
:   431          1442  2
:   432          1443  2
:   433          1444  2 ! Stop metering of this subfunctior
:   434          1445  2 !
:   435          1446  2
:   436          1447  2 PMS_END_SUB ();
:   437          1448  2
:   438          1449  1 END;                                    ! end of routine TRUNCATE

                                        .TITLE  TRUNC
                                        .IDENT  \V04-001\

                                        .EXTRN  PMS_START_SUB, PMS_END_SUB
                                        .EXTRN  FILE_SIZE, SEARCH_FCB
```

```
                                        .EXTRN  CHARGE_QUOTA, DEALLOCATE_BAD
                                        .EXTRN  MARK_DIRTY, CHECKSUM
                                        .EXTRN  GET_MAP_POINTER
                                        .EXTRN  MAKE_POINTER, NEXT_HEADER
                                        .EXTRN  CREATE_BLOCK, INVALIDATE
                                        .EXTRN  INIT_FCB2, WRITE_HEADER
                                        .EXTRN  READ_HEADER, DEL_EXTFCB
                                        .EXTRN  DELETE_FILE

                                        .PSECT  $CODE$,NOWRT,2

                                00BFC 00000     .ENTRY  TRUNCATE, Save R2,R3,R4,R5,R6,R7,R8,R9,R11 ; 1115
                             5E   24   C2 00002         SUBL2   #36, SP
                             52   80   AA 00005         MOVAB   -128(BASE), R2                     ; 1186
                                  08   DD 00009         PUSHL   #8                                 ; 1215
                          0000G CF        01   FB 0000B         CALLS   #1, PMS_START_SUB
                             50   04   AC 00010         MOVQ    FIB, R0                            ; 1217
                                0000V 30 00014         BSBW    TRUNC_CHECKS
                          10   AE   08   AC 00017         MOVL    F:LEHEADER, HEADER                 ; 1224
                          0C   AE   08   AA 0001C         MOVL    8(BASE), FCB                       ; 1225
                    08    AE    0C   AC 00021         SUBL3   #1, TRNVBN, VBN                    ; 1226
                             50   04   AC 00027         MOVL    FIB, R0                            ; 1231
                          1C   A0        01   D0 0002B         MOVL    #1, 28(R0)
                             04   AE   D4 0002F         CLRL    REREAD                             ; 1233
                       51    10   AE        01   C1 00032 1$:     ADDL3   #1, HEADER, R1             ; 1247
                             50        61   9A 00037         MOVZBL  (R1), R0
                             58   10 BE40   3E 0003A         MOVAW   @HEADER[R0], MAP_POINTER
                       51    10   AE        3A   C1 0003F         ADDL3   #58, HEADER, R1            ; 1248
                             50        61   9A 00044         MOVZBL  (R1), R0
                             5B      6840   3E 00047         MOVAW   (MAP_POINTER)[R0], MAP_END
                             59        58   D0 0004B         MOVL    MAP_POINTER, PREV_POINTER  ; 1249
                             6E   08   AE   D0 0004E         MOVL    VBN, HEADER_VBN            ; 1250
                             5B        58   D1 00052 2$:     CMPL    MAP_POINTER, MAP_END       ; 1252
                                       1E   1E 00055         BGEQU   3$
                                0000G  30 00057         BSBW    GET_MAP_POINTER                ; 1254
                          08   AE        56   D1 0005A         CMPL    COUNT, VBN                 ; 1255
                                       47   1E 0005E         BGEQU   5$
                          08   AE        56   C2 00060         SUBL2   COUNT, VBN                 ; 1256
                             50   04   AC   D0 00064         MOVL    FIB, R0                    ; 1257
                          1C   A0        56   C0 00068         ADDL2   COUNT, 28(R0)
                                       56   D5 0006C         TSTL    COUNT                      ; 1258
                                       E2   13 0006E         BEQL    2$
                             59        58   D0 00070         MOVL    MAP_POINTER, PREV_POINTER  ; 1259
                                       DD   11 00073         BRB     2$                         ; 1252
                                0C   AE   DD 00075 3$:     PUSHL   FCB                        ; 126?
                                14   AE   DD 00078         PUSHL   HEADER
                          0000G CF        02   FB 0007B         CALLS   #2, NEXT_HEADER
                             53        50   D0 00080         MOVL    R0, NEW_HEADER
                                       18   13 00083         BEQL    4$                         ; 1268
                          04   AE        01   D0 00085         MOVL    #1, REREAD                 ; 1269
                          10   AE        53   D0 00089         MOVL    NEW_HEADER, HEADER         ; 1270
                                0C   AE   D5 0008D         TSTL    FCB                        ; 1272
                                       A0   13 00090         BEQL    1$
                    50    0C   AE        0C   C1 00092         ADDL3   #12, FCB, R0               ; 1273
                          0C   AE        60   D0 00097         MOVL    (R0), FCB
                                       95   11 0009B         BRB     1$                         ; 1245
                                08   AE   D5 0009D 4$:     TSTL    VBN                        ; 1276
```

```
                                    05  13 000A0          BEQL    5$                                      ;
                              0870  8F  BF 000A2          CHMU    #2160                                   ; 1277
                                    04  000A6          RET                                             ;
                        04    A2    08  AE  CE 000A7 5$:  MNEGL   VBN, 4(R2)                            ; 1286
                              50    C4  AC  D0 000AC      MOVL    FIB, R0                                 ; 1288
                  0C          17    A0  02  E0 000B0      BBS     #2, 23(R0), 6$                          ;
                              50    98  AA  D0 000B5      MOVL    -104(BASE), R0                          ; 1289
                              51    3C  A0  3C 000B9      MOVZWL  60(R0), R1                              ;
                                        51  D7 000BD      DECL    R1                                      ;
                                        02  11 000BF      BRB     7$                                      ;
                                        51  D4 000C1 6$:  CLRL    R1                                      ; 1288
                              51    08  AE  C0 000C3 7$:  ADDL2   VBN, R1                                 ;
                              50    98  AA  D0 000C7      MOVL    -104(BASE), R0                          ; 1292
                              53    3C  A0  3C 000CB      MOVZWL  60(R0), R3                              ;
                              51        53  C6 000CF      DIVL2   R3, R1                                  ;
                              54    3C  A0  3C 000D2      MOVZWL  60(R0), R4                              ;
                  08  AE      51        54  C5 000D6      MULL3   R4, R1, VBN                             ;
                        04    A2    08  AE  C0 000DB      ADDL2   VBN, 4(R2)                              ; 1293
                              50    04  AC  D0 000E0      MOVL    FIB, R0                                 ; 1294
                        1C    A0    08  AE  C0 000E4      ADDL2   VBN, 28(R0)                             ;
                              6E    04  A2  C0 000E9      ADDL2   4(R2), HEADER_VBN                       ; 1295
                              56    08  AE  D1 000ED      CMPL    VBN, COUNT                              ; 1303
                                        20  12 000F1      BNEQ    8$                                      ;
                                    08  AE  D4 000F3      CLRL    VBN                                     ; 1306
                              59        58  D0 000F6      MOVL    MAP_POINTER, PREV_POINTER               ; 1307
                              5B        59  D1 000F9      CMPL    PREV_POINTER, MAP_END                   ; 1308
                                        15  1F 000FC      BLSSU   8$                                      ;
                  50          10    AE      0E  C1 000FE  ADDL3   #14, HEADER, R0                         ; 1309
                                        60  B5 00103      TSTW    (R0)                                    ;
                                        0C  12 00105      BNEQ    8$                                      ;
                  50          10    AE      12  C1 00107  ADDL3   #18, HEADER, R0                         ; 1310
                                        60  B5 0010C      TSTW    (R0)                                    ;
                                        03  12 0010E      BNEQ    8$                                      ;
                                  01A2  31  00110          BRW     21$                                     ;
                              50    04  AC  D0 00113 8$:  MOVL    FIB, R0                                 ; 1321
                  2B          17    A0  02  E1 00117      BBC     #2, 23(R0), 10$                         ;
                              5B        58  D1 0011C      CMPL    MAP_POINTER, MAP_END                    ; 1323
                                        23  12 0011F      BNEQ    9$                                      ;
                        51        56    08  AE  C3 00121  SUBL3   VBN, COUNT, R1                         ; 1324
                              50    98  AA  D0 00126      MOVL    -104(BASE), R0                          ;
  51          3C    A0          10        00  ED 0012A  CMPZV   #0, #16, 60(R0), R1                     ;
                                        12  12 00130      BNEQ    9$                                      ;
                  50          10    AE      0E  C1 00132  ADDL3   #14, HEADER, R0                         ; 1325
                                        60  B5 00137      TSTW    (R0)                                    ;
                                        09  12 00139      BNEQ    9$                                      ;
                  50          10    AE      10  C1 0013B  ADDL3   #16, HEADER, R0                         ; 1326
                                        60  B5 00140      TSTW    (R0)                                    ;
                                        03  13 00142      BEQL    10$                                     ;
                                        14  BF 00144 9$:  CHMU    #20                                     ; 1327
                                    04  00146          RET                                             ;
                              6A        12  88 00147 10$:  BISB2   #18, (BASE)                          ; 1333
                              50    08  AA  D0 0014A      MOVL    8(BASE), R0                             ; 1334
                              51    04  AC  D0 0014E      MOVL    FIB, R1                                 ;
                  38          A0    1C  A1  01  C3 00152  SUBL3   #1, 28(R1), 56(R0)                      ;
                                    12  04  AE  E8 00158  BLBS    REREAD, 11$                             ; 1340
                              50    04  AC  D0 0015C      MOVL    FIB, R0                                 ; 1341
                  50          1C    A0      01  C3 00160  SUBL3   #1, 28(R0), R0                         ;
```

```
                51      10   AE              18  C1 00165          ADDL3   #24, HEADER, R1                                  ;
                61           50              10  9C 0016A          ROTL    #16, R0, (R1)                                    ;
                             7E              01  7D 0016E 11$:     MOVQ    #1, -(SP)                                        ; 1349
                             7E              01  CE 00171          MNEGL   #1, -(SP)                                        ;
                      0000G  CF              03  FB 00174          CALLS   #3, CREATE_BLOCK                                 ;
                        14   AE              50  D0 00179          MOVL    R0, ALT_HEADER                                   ;
                                         14 AE  DD 0017D          PUSHL   ALT_HEADER                                       ; 1350
                      0000G  CF              01  FB 00180          CALLS   #1, INVALIDATE                                   ;
          14    BE      10   BE        0200 8F  28 00185          MOVC3   #512, @HEADER, @ALT_HEADER                       ; 1351
                50           59          10 AE  C3 0018D          SUBL3   HEADER, PREV_POINTER, R0                         ; 1352
                        18   AE        14 BE40  9E 00192          MOVAB   @ALT_HEADER[R0], TRUNC_POINTER                   ;
                             50              02  C6 00198          DIVL2   #2, R0                                           ; 1354
                51      10   AE              3A  C1 0019B          ADDL3   #58, HEADER, R1                                  ;
                52      10   AE              01  C1 001A0          ADDL3   #1, HEADER, R2                                   ;
                61           50              62  83 001A5          SUBB3   (R2), R0, (R1)                                   ;
                                         08 AE  D5 001A9          TSTL    VBN                                              ; 1355
                                            21  13 001AC          BEQL    14$                                              ;
                             58              59  D0 001AE          MOVL    PREV_POINTER, MAP_POINTER                        ; 1358
                                       0000G 30  001B1          BSBW    GET_MAP_POINTER                                  ; 1359
                        C0   8F         01 A9  93 001B4          BITB    1(PREV_POINTER), #192                            ; 1361
                                            05  12 001B9          BNEQ    12$                                              ;
                             7E              69  3C 001BB          MOVZWL  (PREV_POINTER), -(SP)                            ; 1362
                                             02  11 001BE          BRB     13$                                              ;
                                             7E  D4 001C0 12$:     CLRL    -(SP)                                            ; 1361
                                         14 AE  DD 001C2 13$:     PUSHL   HEADER                                           ; 1360
                                             57  DD 001C5          PUSHL   LBN                                              ;
                                         14 AE  DD 001C7          PUSHL   VBN                                              ;
                      0000G  CF              04  FB 001CA          CALLS   #4, MAKE_POINTER                                 ;
                51      10   AE              02  C1 001CF 14$:     ADDL3   #2, HEADER, R1                                   ; 1365
                             50              61  9A 001D4          MOVZBL  (R1), R0                                         ;
                             5B        10 BE40  3E 001D7          MOVAW   @HEADER[R0], MAP_END                             ;
                             59              5B  D1 001DC          CMPL    MAP_END, PREV_POINTER                            ; 1366
                                             09  15 001DF          BLEQ    15$                                              ;
                             5B              59  C2 001E1          SUBL2   PREV_POINTER, R11                                ; 1367
          5B        00                       6E  00  2C 001E4      MOVC5   #0, (SP), #0, R11, (PREV_POINTER)                ;
                                                 69     001E9                                                               ;
                50      10   AE              04  C1 001EA 15$:     ADDL3   #4, HEADER, R0                                   ; 1369
                             56              60  3C 001EF          MOVZWL  (R0), EX_SEGNUM                                  ;
                                             56  D6 001F2          INCL    EX_SEGNUM                                        ;
                57      10   AE              0E  C1 001F4          ADDL3   #14, HEADER, R7                                  ; 1370
            1C  AE           67              06  28 001F9          MOVC3   #6, (R7), EXT_FID                                ;
                57      10   AE              0E  C1 001FE          ADDL3   #14, HEADER, R7                                  ; 1371
    06          00           6E              00  2C 00203          MOVC5   #0, (SP), #0, #6, (R7)                           ;
                                                 67     00208                                                               ;
                                         10 AE  DD 00209          PUSHL   HEADER                                           ; 1372
                      0000G  CF              01  FB 0020C          CALLS   #1, CHECKSUM                                     ;
                      0000G  CF              00  FB 00211          CALLS   #0, WRITE_HEADER                                 ; 1373
                                         0C AE  D5 00216          TSTL    FCB                                              ; 1375
                                            12  13 00219          BEQL    16$                                              ;
                        08   AA         0C AE  D1 0021B          CMPL    FCB, 8(BASE)                                     ;
                                            0B  13 00220          BEQL    16$                                              ;
                                         10 AE  DD 00222          PUSHL   HEADER                                           ; 1376
                                         10 AE  DD 00225          PUSHL   FCB                                              ;
                      0000G  CF              02  FB 00228          CALLS   #2, INIT_FCB2                                    ;
                18           6A             1D  E0 0022D 16$:     BBS     #29, (BASE), 17$                                 ; 1383
                                         14 AE  DD 00231          PUSHL   ALT_HEADER                                       ; 1386
                      0000G  CF              01  FB 00234          CALLS   #1, FILE_SIZE                                    ;
```

```
                                    02  DD 00239          PUSHL   #2                                 ; 1388
          7E       04  AE           50  C3 0023B          SUBL3   HEADER_SIZE, HEADER_VBN, -(SP)     ; 1387
          52       18  AE           3C  C1 00240          ADDL3   #60, HEADER, R2
                                    62  DD 00245          PUSHL   (R2)
               0000G   CF           03  FB 00247          CALLS   #3, CHARGE_QUOTA
                       52      04   AE  D0 0024C 17$:     MOVL    REREAD, REREAD2                    ; 1391
                       50      04   AC  D0 00250          MOVL    FIB, R0                            ; 1397
          18       17  A0           02  E1 00254          BBC     #2, 23(R0), 18$
                               08   AE  DD 00259          PUSHL   VBN                                ; 1400
                               1C   AE  DD 0025C          PUSHL   TRUNC_POINTER
                               1C   AE  DD 0025F          PUSHL   ALT_HEADER
                                    50  DD 00262          PUSHL   R0
           00000000G   00           04  FB 00264          CALLS   #4, DEALLOCATE_BAD
                   04  AE           01  D0 0026B          MOVL    #1, REREAD                         ; 1411
                                    10  11 0026F          BRB     19$                                ; 1397
                               08   AE  DD 00271 18$:     PUSHL   VBN                                ; 1414
                               1C   AE  DD 00274          PUSHL   TRUNC_POINTER
                               1C   AE  DD 00277          PUSHL   ALT_HEADER
                                    50  DD 0027A          PUSHL   R0
               0000V   CF           04  FB 0027C          CALLS   #4, TRUNCATE_HEADER
                               1C   AE  B5 00281 19$:     TSTW    EXT_FID                            ; 1416
                                    05  12 00284          BNEQ    20$
                               20   AE  B5 00286          TSTW    EXT_FID+4                          ; 1417
                                    2A  13 00289          BEQL    21$
                   04  AE           01  D0 0028B 20$:     MOVL    #1, REREAD                         ; 1420
                                    56  DD 0028F          PUSHL   EX_SEGNUM                          ; 1421
                               20   AE  9F 00291          PUSHAB  EXT_FID
                               14   AE  DD 00294          PUSHL   FCB
                                    7E  D4 00297          CLRL    -(SP)
               0000G   CF           04  FB 00299          CALLS   #4, NEXT_HEADER
                   10  AE           50  D0 0029E          MOVL    R0, HEADER
                               0C   AE  DD 002A2          PUSHL   FCB                                ; 1422
               0000G   CF           01  FB 002A5          CALLS   #1, DEL_EXTFCB
                               10   AE  DD 002AA          PUSHL   HEADER                             ; 1423
                               04   AC  DD 002AD          PUSHL   FIB
               0000G   CF           02  FB 002B0          CALLS   #2, DELETE_FILE
                       0E      04   AE  E9 002B5 21$:     BLBC    REREAD, 22$                        ; 1433
                               08   AA  DD 002B9          PUSHL   8(BASE)                            ; 1434
                                    7E  D4 002BC          CLRL    -(SP)
               0000G   CF           02  FB 002BE          CALLS   #2, READ_HEADER
                   10  AE           50  D0 002C3          MOVL    R0, HEADER
                       1A           52  E9 002C7 22$:     BLBC    REREAD2, 23$                       ; 1436
                       50      04   AC  D0 002CA          MOVL    FIB, R0                            ; 1439
          50       1C  A0           01  C3 002CE          SUBL3   #1, 28(R0), R0
          51       10  AE           18  C1 002D3          ADDL3   #24, HEADER, R1
          61           50           10  9C 002D8          ROTL    #16, R0, (R1)
                               10   AE  DD 002DC          PUSHL   HEADER                             ; 1440
               0000G   CF           01  FB 002DF          CALLS   #1, MARK_DIRTY
               0000G   CF           00  FB 002E4 23$:     CALLS   #0, PMS_END_SUB                    ; 1447
                                        04 002E9          RET                                        ; 1449

; Routine Size:  746 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  440     1450  1 GLOBAL ROUTINE TRUNCATE_HEADER (FIB, HEADER, POINTER, LAST_COUNT) : L_NORM NOVALUE =
;  441     1451  1
;  442     1452  1 !++
;  443     1453  1 !
;  444     1454  1 ! FUNCTIONAL DESCRIPTION:
;  445     1455  1 !
;  446     1456  1 !       This routine returns the indicated retrieval pointers in the given
;  447     1457  1 !       file header to the storage map.
;  448     1458  1 !
;  449     1459  1 !
;  450     1460  1 ! CALLING SEQUENCE:
;  451     1461  1 !       TRUNCATE_HEADER (ARG1, ARG2, ARG3, ARG4)
;  452     1462  1 !
;  453     1463  1 ! INPUT PARAMETERS:
;  454     1464  1 !       ARG1: address of FIB of operation
;  455     1465  1 !       ARG2: address of file header
;  456     1466  1 !       ARG3: address of first retrieval pointer to process, if present
;  457     1467  1 !       ARG4: new count field of first pointer, if present
;  458     1468  1 !
;  459     1469  1 ! IMPLICIT INPUTS:
;  460     1470  1 !       NONE
;  461     1471  1 !
;  462     1472  1 ! OUTPUT PARAMETERS:
;  463     1473  1 !       NONE
;  464     1474  1 !
;  465     1475  1 ! IMPLICIT OUTPUTS:
;  466     1476  1 !       NONE
;  467     1477  1 !
;  468     1478  1 ! ROUTINE VALUE:
;  469     1479  1 !       NONE
;  470     1480  1 !
;  471     1481  1 ! SIDE EFFECTS:
;  472     1482  1 !       file header altered, storage map altered
;  473     1483  1 !
;  474     1484  1 !--
;  475     1485  1
;  476     1486  2 BEGIN
;  477     1487  2
;  478     1488  2 MAP
;  479     1489  2         FIB             : REF BBLOCK,   ! user FIB
;  480     1490  2         HEADER          : REF BBLOCK;   ! file header
;  481     1491  2
;  482     1492  2 GLOBAL REGISTER
;  483     1493  2         COUNT           = 6,            ! count of blocks returned
;  484     1494  2         LBN             = 7,            ! LBN of map entry
;  485     1495  2         MAP_POINTER     = 8 : REF BBLOCK; ! pointer to scan map
;  486     1496  2
;  487     1497  2 LOCAL
;  488     1498  2         MAP_END;                        ! address of end of map area
;  489     1499  2
;  490     1500  2 BIND_COMMON;
;  491     1501  2
;  492     1502  2 EXTERNAL ROUTINE
;  493     1503  2         GET_MAP_POINTER : L_MAP_POINTER, ! get value of next map entry
;  494     1504  2         RETURN_BLOCKS   : L_NORM ADDRESSING_MODE (GENERAL);
;  495     1505  2                                         ! return blocks to storage map
;  496     1506  2
```

```
;  497          1507  2
;  498          1508  2 LOCAL
;  499          1509  2         ERASE_FLAG;
;  500          1510  2
;  501          1511  2 ! Determine if blocks being returned should be erased.  Erase them if
;  502          1512  2 ! either the volume or file erase attribute is set.
;  503          1513  2 !
;  504          1514  2 ERASE_FLAG = 0;                          ! Assume no erase necessary
;  505          1515  2 IF .CURRENT_VCB[VCB$V_ERASE]             ! Check the volume attribute
;  506          1516  2 OR .HEADER[FH2$V_ERASE]                  ! Check the file attribute in the header
;  507          1517  2 THEN
;  508          1518  2     ERASE_FLAG = 1
;  509          1519  2 ELSE
;  510          1520  2     IF .PRIMARY_FCB NEQ 0               ! Check the file attribute in the FCB
;  511          1521  2     THEN
;  512          1522  2         IF .PRIMARY_FCB[FCB$V_ERASE]
;  513          1523  2         THEN
;  514          1524  2             ERASE_FLAG = 1;
;  515          1525  2
;  516          1526  2 !
;  517          1527  2 ! Establish pointers into the file header. If explicit args are supplied, use
;  518          1528  2 ! them; else default to releasing the entire file header.
;  519          1529  2 !
;  520          1530  2
;  521          1531  2 MAP_POINTER = .HEADER + .HEADER[FH2$B_MPOFFSET]*2;
;  522          1532  2 MAP_END = .MAP_POINTER + .HEADER[FH2$B_MAP_INUSE]*2;
;  523          1533  2
;  524          1534  2 IF ACTUALCOUNT GEQ 4
;  525          1535  2 THEN
;  526          1536  3     BEGIN
;  527          1537  3     MAP_POINTER = .POINTER;
;  528          1538  3     IF .LAST_COUNT NEQ 0
;  529          1539  3     THEN
;  530          1540  4         BEGIN
;  531          1541  4         GET_MAP_POINTER ();
;  532          1542  4         RETURN_BLOCKS (.LBN+.LAST_COUNT, .COUNT-.LAST_COUNT, .ERASE_FLAG);
;  533          1543  4         FIB[FIB$L_EXSZ] = .FIB[FIB$L_EXSZ] + .COUNT - .LAST_COUNT;
;  534          1544  3         END;
;  535          1545  2     END;
;  536          1546  2
;  537          1547  2 ! Now scan the map area, cleaning out pointers and releasing blocks.
;  538          1548  2 !
;  539          1549  2
;  540          1550  2 UNTIL .MAP_POINTER GEQA .MAP_END DO
;  541          1551  3     BEGIN
;  542          1552  3     GET_MAP_POINTER ();
;  543          1553  3     RETURN_BLOCKS (.LBN, .COUNT, .ERASE_FLAG);
;  544          1554  3     FIB[FIB$L_EXSZ] = .FIB[FIB$L_EXSZ] + .COUNT;
;  545          1555  2     END;
;  546          1556  2
;  547          1557  1 END;                                     ! end of routine TRUNCATE_HEADER


                                                        .EXTRN  RETURN_BLOCKS

                                        01DC 00000      .ENTRY  TRUNCATE_HEADER, Save R2,R3,R4,R6,R7,R8     : 1450
```

```
                 54 00000000G  00  9E 00002        MOVAB   RETURN_BLOCKS, R4              ;
                               52  D4 00009        CLRL    ERASE_FLAG                     ; 1514
                 50       98   AA  D0 0000B        MOVL    -104(BASE), R0                 ; 1515
      14         53   A0       03  E0 0000F        BBS     #3, 83(R0), 1$                 ;
                 50       08   AC  D0 00014        MOVL    HEADER, R0                     ; 1516
      0B         36   A0       01  E0 00018        BBS     #1, 54(R0), 1$                 ;
                 50       08   AA  D0 0001D        MOVL    8(BASE), R0                    ; 1520
                               08  13 00021        BEQL    2$                             ;
      03         22   A0       06  E1 00023        BBC     #6, 34(R0), 2$                 ; 1522
                      52       01  D0 00028 1$:    MOVL    #1, ERASE_FLAG                 ; 1524
                      51  08   AC  D0 0002B 2$:    MOVL    HEADER, R1                     ; 1531
                      50  01   A1  9A 0002F        MOVZBL  1(R1), R0                      ;
                      58     6140  3E 00033        MOVAW   (R1)[R0], MAP_POINTER          ;
                      50  3A   A1  9A 00037        MOVZBL  58(R1), R0                     ; 1532
                      53     6840  3E 0003B        MOVAW   (MAP_POINTER)[R0], MAP_END     ;
                      04       6C  91 0003F        CMPB    (AP), #4                       ; 1534
                               29  1F 00042        BLSSU   3$                             ;
                 58       0C   AC  D0 00044        MOVL    POINTER, MAP_POINTER           ; 1537
                          10   AC  D5 00048        TSTL    LAST_COUNT                     ; 1538
                               20  13 0004B        BEQL    3$                             ;
                            0000G  30 0004D        BSBW    GET_MAP_POINTER                ; 1541
                               52  DD 00050        PUSHL   ERASE_FLAG                     ; 1542
      7E         56       10   AC  C3 00052        SUBL3   LAST_COUNT, COUNT, -(SP)       ;
                          10 BC47  9F 00057        PUSHAB  @LAST_COUNT[LBN]               ;
                      64       03  FB 0005B        CALLS   #3, RETURN_BLOCKS              ;
                      50  04   AC  D0 0005E        MOVL    FIB, R0                        ; 1543
      51              56  18   A0  C1 00062        ADDL3   24(R0), COUNT, R1              ;
  18  A0              51  10   AC  C3 00067        SUBL3   LAST_COUNT, R1, 24(R0)         ;
                      53       58  D1 0006D 3$:    CMPL    MAP_POINTER, MAP_END           ; 1550
                               16  1E 00070        BGEQU   4$                             ;
                            0000G  30 00072        BSBW    GET_MAP_POINTER                ; 1552
                               52  DD 00075        PUSHL   ERASE_FLAG                     ; 1553
                               56  DD 00077        PUSHL   COUNT                          ;
                               57  DD 00079        PUSHL   LBN                            ;
                      64       03  FB 0007B        CALLS   #3, RETURN_BLOCKS              ;
                      50  04   AC  D0 0007E        MOVL    FIB, R0                        ; 1554
                 18   A0       56  C0 00082        ADDL2   COUNT, 24(R0)                  ;
                               E5  11 00086        BRB     3$                             ; 1550
                                   04 00088 4$:    RET                                    ; 1557

; Routine Size:  137 bytes,    Routine Base:  $CODE$ + 02EA


;  548          1558  1
```

```
1 GLOBAL ROUTINE TRUNC_CHECKS (FIB, HEADER) : L_JSB_2ARGS NOVALUE =
1
1 !
1 !
1 !
1
2 BEGIN
2
2 MAP
2         FIB      : REF BBLOCK,
2         HEADER   : REF BBLOCK;
2
2 BIND_COMMON;
2
2 ! The block count must be zero 'default).
2 ! If the operation calls for the blocks to be turned over to the bad block
2 ! file, the caller must be system.
2 !
2
2 IF .FIB[FIB$V_MARKBAD]
2 AND NOT .CLEANUP_FLAGS[CLF_SYSPRV]
2 THEN ERR_EXIT (SS$_NOPRIV);
2
2 ! Check for the index file INDEXF.SYS
2 !
2
2 IF  .HEADER[FH2$W_FID_NUM] EQL FID$C_INDEXF
2 AND .HEADER[FH2$W_FID_SEQ] EQL FID$C_INDEXF
2 AND .HEADER[FH2$B_FID_NMX] EQL 0
2 THEN ERR_EXIT (SS$_NOPRIV);
2
2 IF .FIB[FIB$L_EXSZ] NEQ 0 THEN ERR_EXIT (SS$_BADPARAM);
2
2 ! Init the user's return parameters.
2 !
2
2 FIB[FIB$L_EXSZ] = 0;
2
1 END;
```

```
04      17  A0          02  E1 00000 TRUNC_CHECKS::
                                              BBC     #2, 23(FIB), 1$        ; 1578
            11      01  AA  E9 00005          BLBC    1(BASE), 2$            ; 1579
            01      08  A1  B1 00009 1$:      CMPW    8(HEADER), #1          ; 1585
                        0E  12 0000D          BNEQ    3$                     ;
            01      0A  A1  B1 0000F          CMPW    10(HEADER), #1         ; 1586
                        08  12 00013          BNEQ    3$                     ;
                    0D  A1  95 00015          TSTB    13(HEADER)             ; 1587
                        03  12 00018          BNEQ    3$                     ;
                        24  BF 0001A 2$:      CHMU    #36                    ; 1588
                            05 0001C          RSB                            ;
                    18  A0  D5 0001D 3$:      TSTL    24(FIB)                ; 1590
                        03  13 00020          BEQL    4$                     ;
```

```
                                     14  BF 00022          CHMU    #20                                  ;
                                         05 00024          RSB                                          ;
                                 18  AC  D4 00025 4$:      CLRL    24(FIB)                              ; 1595
                                         05 00028          RSB                                          ; 1597

; Routine Size:  41 bytes,    Routine Base:  $CODE$ + 0373

;  589          1598  1
;  590          1599  1 END
;  591          1600  0 ELUDOM



;                                  PSECT SUMMARY
;
;       Name                     Bytes                          Attributes
;
;  $CODE$                         924  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                               Library Statistics
;
;                                 -------- Symbols --------      Pages       Processing
;       File                      Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA18:[SYSLIB]LIB.L32;1   18619        49         0        1000        00:02.0



;                                    COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:TRUNC/OBJ=OBJ$:TRUNC MSRC$:TRUNC/UPDATE=(BUG$:TRUNC)

; Size:          924 code + 0 data bytes
; Run Time:        00:45.2
; Elapsed Time:    01:09.5
; Lines/CPU Min:    2123
; Lexemes/CPU-Min: 49512
; Memory Used:  353 pages
; Compilation Complete
```

0444 AH-EF71A-SE
VAX/VMS V4.1 SRC LST MCRF UPD