```
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
```

FILEID**QUOTAUTIL

```
  QQQQQQ    UU      UU    OOOOOO    TTTTTTTTTT    AAAAAA    UU      UU  TTTTTTTTTT    IIIIII    LL
  QQQQQQ    UU      UU    OOOOOO    TTTTTTTTTT    AAAAAA    UU      UU  TTTTTTTTTT    IIIIII    LL
QQ      QQ  UU      UU  OO      OO      TT      AA      AA  UU      UU      TT          II      LL
QQ      QQ  UU      UU  OO      OO      TT      AA      AA  UU      UU      TT          II      LL
QQ      QQ  UU      UU  OO      OO      TT      AA      AA  UU      UU      TT          II      LL
QQ      QQ  UU      UU  OO      OO      TT      AA      AA  UU      UU      TT          II      LL
QQ      QQ  UU      UU  OO      OO      TT      AA      AA  UU      UU      TT          II      LL
QQ      QQ  UU      UU  OO      OO      TT      AA      AA  UU      UU      TT          II      LL
QQ  QQ  QQ  UU      UU  OO      OO      TT      AAAAAAAAAA  UU      UU      TT          II      LL
QQ  QQ  QQ  UU      UU  OO      OO      TT      AAAAAAAAAA  UU      UU      TT          II      LL
QQ    QQ    UU      UU  OO      OO      TT      AA      AA  UU      UU      TT          II      LL              ....
QQ    QQ    UU      UU  OO      OO      TT      AA      AA  UU      UU      TT          II      LL              ....
  QQQQ  QQ  UUUUUUUUUU    OOOOOO        TT      AA      AA  UUUUUUUUUU      TT        IIIIII    LLLLLLLLLL      ....
  QQQQ  QQ  UUUUUUUUUU    OOOOOO        TT      AA      AA  UUUUUUUUUU      TT        IIIIII    LLLLLLLLLL      ....

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 QUOTAUTIL (
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 IDENT = 'V04-001'
;     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 module contains routines that inplement the ACP control
;    38          0038  1 !       functions that operate on the quota file.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:
;    41          0041  1 !
;    42          0042  1 !       STARLET operating system, including privileged system services
;    43          0043  1 !       and internal exec routines.
;    44          0044  1 !
;    45          0045  1 !--
;    46          0046  1 !
;    47          0047  1 !
;    48          0048  1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  31-May-1979  15:18
;    49          0049  1 !
;    50          0050  1 ! MODIFIED BY:
;    51          0051  1 !
;    52          0052  1 !       V04-001 ACG0466         Andrew C. Goldstein,    12-Sep-1984  14:38
;    53          0053  1 !               Flush quota file blocks from cache when disabling quotas
;    54          0054  1 !
;    55          0055  1 !       V03-012 CDS0008         Christian D. Saether    29-Aug-1984
;    56          0056  1 !               Deal with potential multi-header quota file caused
;    57          0057  1 !               by ACL's.
```

```
;    58          0058  1 !
;    59          0059  1 !       V03-011 CDS0007         Christian D. Saether    23-Aug-1984
;    60          0060  1 !               Mark quota fcb stale clusterwide when it is extended.
;    61          0061  1 !
;    62          0062  1 !       V03-010 ACG0438         Andrew C. Goldstein,    18-Jul-1984  20:32
;    63          0063  1 !               Implement quota cache lock; dequeue when cache is released.
;    64          0064  1 !               Use central dequeue routine.
;    65          0065  1 !
;    66          0066  1 !       V03-009 CDS0006         Christian D. Saether     9-May-1984
;    67          0067  1 !               Add serialization call to flush_quo_cache routine.
;    68          0068  1 !
;    69          0069  1 !       V03-008 CDS0005         Christian D. Saether    19-Apr-1984
;    70          0070  1 !               Bump REFCNT in quota file fcb also.
;    71          0071  1 !
;    72          0072  1 !       V03-007 ACG0412         Andrew C. Goldstein,    22-Mar-1984  18:35
;    73          0073  1 !               Implement agent access mode support; add access mode to
;    74          0074  1 !               protection check call
;    75          0075  1 !
;    76          0076  1 !       V03-006 ACG0400         Andrew C. Goldstein,    7-Mar-1984  17:07
;    77          0077  1 !               Implement cluster-wide quota cache, remove marking
;    78          0078  1 !               of SCB for quotas.
;    79          0079  1 !
;    80          0080  1 !       V03-005 CDS0004         Christian D. Saether     1-Mar-1984
;    81          0081  1 !               Remove reference to FLUSH_FID.
;    82          0082  1 !
;    83          0083  1 !       V03-004 CDS0003         Christian D. Saether    30-Dec-1983
;    84          0084  1 !               Use L_NORM linkage and BIND_COMMON macro.
;    85          0085  1 !
;    86          0086  1 !       V03-003 CDS0002         Christian D. Saether     6-Dec-1983
;    87          0087  1 !               Volume lock check on quota file modification request
;    88          0088  1 !               has changed.  NOALLOC is no longer set.
;    89          0089  1 !
;    90          0090  1 !       V03-002 CDS0001         Christian D. Saether    17-Oct-1983
;    91          0091  1 !               Add minimal quota checking support for xqp.
;    92          0092  1 !
;    93          0093  1 !       V03-001 ACG0308         Andrew C. Goldstein,    14-Jan-1983  14:26
;    94          0094  1 !               Fix consistency problems in linking FCB's
;    95          0095  1 !
;    96          0096  1 !       V02-005 ACG0213         Andrew C. Goldstein,    13-Aug-1981  13:42
;    97          0097  1 !               Remove write lock from quota file
;    98          0098  1 !
;    99          0099  1 !       V02-004 ACG0167         Andrew C. Goldstein,    16-Apr-1980  19:27
;   100          0100  1 !               Previous revision history moved to F11B.REV
;   101          0101  1 !**
;   102          0102  1
;   103          0103  1
;   104          0104  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;   105          0105  1 REQUIRE 'SRC$:FCPDEF.B32';
;   106          1096  1
;   107          1097  1
;   108          1098  1 FORWARD ROUTINE
;   109          1099  1         QUOTA_FILE_OP   : L_NORM NOVALUE, ! general quota file operations
;   110          1100  1         FLUSH_QUO_CACHE : L_NORM NOVALUE, ! flush dirty entries from quota cache
;   111          1101  1         DEACC_QFILE     : L_NORM,         ! deaccess the quota file
;   112          1102  1         RET_QENTRY      : L_NORM,         ! return quota file entry to user
;   113          1103  1         CONN_QFILE      : L_NORM NOVALUE, ! connect the quota file
;   114          1104  1         MAKE_QFCB       : L_NORM;         ! complete quota file access
```

```
;  116          1105  1 GLOBAL ROUTINE QUOTA_FILE_OP (ABD, FIB) : L_NORM NOVALUE =
;  117          1106  1
;  118          1107  1 !++
;  119          1108  1 !
;  120          1109  1 ! FUNCTIONAL DESCRIPTION:
;  121          1110  1 !
;  122          1111  1 !       This routine implements most of the quota file ACP control functions
;  123          1112  1 !       (i.e., the ones that are performed on the open quota file).
;  124          1113  1 !
;  125          1114  1 ! CALLING SEQUENCE:
;  126          1115  1 !       QUOTA_FILE_OP (ARG1, ARG2)
;  127          1116  1 !
;  128          1117  1 ! INPUT PARAMETERS:
;  129          1118  1 !       ARG1: address of buffer descriptor packet
;  130          1119  1 !       ARG2: address of user FIB
;  131          1120  1 !
;  132          1121  1 ! IMPLICIT INPUTS:
;  133          1122  1 !       CLEANUP_FLAGS: cleanup action and status flags
;  134          1123  1 !       CURRENT_VCB: VCB of current volume
;  135          1124  1 !       IO_PACKET: I/O packet being processed
;  136          1125  1 !       QUOTA_RECORD: record number of found quota file record
;  137          1126  1 !       FREE_QUOTA: record number of first free quota file record
;  138          1127  1 !
;  139          1128  1 ! OUTPUT PARAMETERS:
;  140          1129  1 !       NONE
;  141          1130  1 !
;  142          1131  1 ! IMPLICIT OUTPUTS:
;  143          1132  1 !       PRIMARY_FCB: FCB of quota file
;  144          1133  1 !
;  145          1134  1 ! ROUTINE VALUE:
;  146          1135  1 !       NONE
;  147          1136  1 !
;  148          1137  1 ! SIDE EFFECTS:
;  149          1138  1 !       quota file searched, modified, etc.
;  150          1139  1 !
;  151          1140  1 !--
;  152          1141  1
;  153          1142  2 BEGIN
;  154          1143  2
;  155          1144  2 MAP
;  156          1145  2         ABD             : REF BBLOCKVECTOR [,ABD$C_LENGTH],
;  157          1146  2                                   ! buffer descriptor vector
;  158          1147  2         FIB             : REF BBLOCK;   ! user FIB
;  159          1148  2
;  160          1149  2 LITERAL
;  161          1150  2         RECS_PER_BLOCK  = 512 / DQF$C_LENGTH,
;  162          1151  2
;  163          1152  2         MAX_QFUNC       = MAXU (FIB$C_DSA_QUOTA,
;  164          1153  2                                 FIB$C_EXA_QUOTA,
;  165          1154  2                                 FIB$C_REM_QUOTA,
;  166          1155  2                                 FIB$C_MOD_QUOTA,
;  167          1156  2                                 FIB$C_ADD_QUOTA
;  168          1157  2                                 ),
;  169          1158  2
;  170          1159  2         MIN_QFUNC       = MINU (FIB$C_DSA_QUOTA,
;  171          1160  2                                 FIB$C_EXA_QUOTA,
;  172          1161  2                                 FIB$C_REM_QUOTA,
```

```
;  173          1162  2                                FIB$C_MOD_QUOTA,
;  174          1163  2                                FIB$C_ADD_QUOTA
;  175          1164  2                                );
;  176          1165  2
;  177          1166  2 LOCAL
;  178          1167  2         TEMP1,                          ! random temp storage
;  179          1168  2         TEMP2,                          ! more of the same
;  180          1169  2         FCB             : REF BBLOCK,   ! address of quota file FCB
;  181          1170  2         BUFFER          : REF BBLOCK,   ! disk block buffer
;  182          1171  2         Q_RECORD        : REF BBLOCK,   ! record found in quota file
;  183          1172  2         Q_BLOCK         : REF BBLOCK;   ! quota arg block from user
;  184          1173  2
;  185          1174  2 BIND_COMMON;
;  186          1175  2
;  187          1176  2 EXTERNAL ROUTINE
;  188          1177  2         MAKE_FCB_STALE  : L_NORM NOVALUE, ! mark fcb stale clusterwide
;  189          1178  2         SERIAL_FILE     : L_NORM,       ! serialize on given file
;  190          1179  2         ALLOCATION_LOCK : L_NORM,       ! serialize on volume allocation
;  191          1180  2         SWITCH_VOLUME   : L_NORM,       ! switch volume context
;  192          1181  2         SEARCH_QUOTA    : L_NORM,       ! find entry in quota file
;  193          1182  2         CHECK_PROTECT   : L_NORM,       ! check file protection
;  194          1183  2         GET_QUOTA_LOCK  : L_NORM,       ! take lock on quota cache entry
;  195          1184  2         REL_QUOTA_LOCK  : L_NORM,       ! release lock on quota cache entry
;  196          1185  2         WRITE_DIRTY     : L_NORM NOVALUE, ! write dirty buffers
;  197          1186  2         READ_BLOCK      : L_NORM,       ! read a disk block
;  198          1187  2         EXTEND_CONTIG   : L_NORM,       ! extend a contiguous file
;  199          1188  2         WRITE_QUOTA     : L_NORM;       ! write quota file record
;  200          1189  2
;  201          1190  2
;  202          1191  2 ! Do the preliminary setup and validation. All operations handled by this
;  203          1192  2 ! routine operate on RVN 1 of a volume set and require the quota file to
;  204          1193  2 ! be connected.
;  205          1194  2 !
;  206          1195  2
;  207          1196  2 SWITCH_VOLUME (1);
;  208          1197  2 PRIMARY_FCB = FCB = .CURRENT_VCB[VCB$L_QUOTAFCB];
;  209          1198  2 IF .FCB EQL 0
;  210          1199  2 THEN ERR_EXIT (SS$_QFNOTACT);
;  211          1200  2
;  212          1201  2 SERIAL_FILE (FCB [FCB$W_FID]);
;  213          1202  2
;  214          1203  2 ALLOCATION_LOCK ();
;  215          1204  2
;  216          1205  2 ! Do additional validation which is common for several functions. All but
;  217          1206  2 ! the disable function require a quota file search and require the quota
;  218          1207  2 ! argument block (P2) to be present.
;  219          1208  2 !
;  220          1209  2
;  221          1210  2 IF .FIB[FIB$W_CNTRLFUNC] NEQ FIB$C_DSA_QUOTA
;  222          1211  2 THEN
;  223          1212  3     BEGIN
;  224          1213  3     IF .ABD[ABD$C_NAME, ABD$W_COUNT] LSSU DQF$C_LENGTH
;  225          1214  3     THEN ERR_EXIT (SS$_INSFARG);
;  226          1215  3     Q_BLOCK = ABD[ABD$C_NAME, ABD$W_TEXT] + .ABD[ABD$C_NAME, ABD$W_TEXT] + 1;
;  227          1216  3
;  228          1217  3     Q_RECORD = SEARCH_QUOTA (.Q_BLOCK[DQF$L_UIC], .FIB[FIB$L_CNTRLVAL], .FIB[FIB$L_WCC], 0);
;  229          1218  3     IF .FIB[FIB$V_ALL_MEM]
```

```
;  230          1219  3     OR .FIB[FIB$V_ALL_GRP]
;  231          1220  3     THEN FIB[FIB$L_WCB] = .QUOTA_RECORD;
;  232          1221  3
;  233          1222  3 ! All functions except disable and examine require write access to the
;  234          1223  3 ! quota file; examine requires read access except when examining one's
;  235          1224  3 ! own quota.
;  236          1225  3 !
;  237          1226  3
;  238          1227  3     IF .FIB[FIB$W_CNTRLFUNC] NEQ FIB$C_EXA_QUOTA
;  239          1228  3     THEN
;  240          1229  3         CHECK_PROTECT (WRITE_ACCESS, 0, .FCB, 0)
;  241          1230  3     ELSE
;  242          1231  4         BEGIN
;  243          1232  4         IF .FIB[FIB$V_ALL_MEM]
;  244          1233  4         OR .FIB[FIB$V_ALL_GRP]
;  245          1234  4         OR .Q_BLOCK[DQF$L_UIC] NEQ
;  246          1235  4             .BBLOCK [.IO_PACKET[IRP$L_ARB], ARB$L_UIC]
;  247          1236  4         THEN CHECK_PROTECT (READ_ACCESS, 0, .FCB, 0);
;  248          1237  3         END;
;  249          1238  3
;  250          1239  3 ! All functions except disable and add require the quota file search to be
;  251          1240  3 ! successful.
;  252          1241  3 !
;  253          1242  3
;  254          1243  3     IF .FIB[FIB$W_CNTRLFUNC] NEQ FIB$C_ADD_QUOTA
;  255          1244  3     THEN
;  256          1245  3         IF .Q_RECORD EQL 0
;  257          1246  3         THEN ERR_EXIT (SS$_NODISKQUOTA);
;  258          1247  2     END;
;  259          1248  2
;  260          1249  2 ! Dispatch on the function and do it.
;  261          1250  2 !
;  262          1251  2
;  263          1252  2 CASE .FIB[FIB$W_CNTRLFUNC] FROM MIN_QFUNC TO MAX_QFUNC OF
;  264          1253  2     SET
;  265          1254  2
;  266          1255  2     [FIB$C_DSA_QUOTA]:                  ! disable disk quotas
;  267          1256  3         BEGIN
;  268          1257  3         IF NOT .CLEANUP_FLAGS[CLF_SYSPRV]
;  269          1258  3         THEN ERR_EXIT (SS$_NOPRIV);
;  270          1259  3         FLUSH_QUO_CACHE ();
;  271          1260  3         WRITE_DIRTY (-1);
;  272          1261  3         KERNEL_CALL (DEACC_QFILE);
;  273          1262  2         END;
;  274          1263  2
;  275          1264  2     [FIB$C_EXA_QUOTA]:                  ! examine quota file entry
;  276          1265  3         BEGIN
;  277          1266  3         KERNEL_CALL (RET_QENTRY, .Q_RECORD, .ABD);
;  278          1267  2         END;
;  279          1268  2
;  280          1269  2     [FIB$C_REM_QUOTA]:                  ! remove quota file entry
;  281          1270  3         BEGIN
;  282          1271  3         IF .Q_RECORD[DQF$L_USAGE] NEQ 0
;  283          1272  3         THEN ERR_STATUS (SS$_OVRDSKQUOTA);
;  284          1273  3         KERNEL_CALL (RET_QENTRY, .Q_RECORD, .ABD);
;  285          1274  3         GET_QUOTA_LOCK (.QUOTA_INDEX, LCK$K_EXMODE);
;  286          1275  3         CH$FILL (0, DQF$C_LENGTH, .Q_RECORD);
```

```
;  287          1276  3          WRITE_QUOTA (.Q_RECORD);
;  288          1277  3          REL_QUOTA_LOCK (.QUOTA_INDEX);
;  289          1278  2          END;
;  290          1279  2
;  291          1280  2      [FIB$C_MOD_QUOTA]:                  ! modify quota file entry
;  292          1281  3          BEGIN
;  293          1282  3          IF .FIB[FIB$V_MOD_USE]
;  294          1283  3          THEN
;  295          1284  4              BEGIN
;  296          1285  4              IF .BLOCK_LOCKID EQL 0
;  297          1286  4              THEN ERR_EXIT (SS$_ACCONFLICT);
;  298          1287  4              Q_RECORD[DQF$L_USAGE] = .Q_BLOCK[DQF$L_USAGE];
;  299          1288  3              END;
;  300          1289  3          IF .FIB[FIB$V_MOD_PERM]
;  301          1290  3          THEN
;  302          1291  3              Q_RECORD[DQF$L_PERMQUOTA] = .Q_BLOCK[DQF$L_PERMQUOTA];
;  303          1292  3          IF .FIB[FIB$V_MOD_OVER]
;  304          1293  3          THEN
;  305          1294  3              Q_RECORD[DQF$L_OVERDRAFT] = .Q_BLOCK[DQF$L_OVERDRAFT];
;  306          1295  3          IF .Q_RECORD[DQF$L_USAGE] GTRU .Q_RECORD[DQF$L_PERMQUOTA]
;  307          1296  3          THEN ERR_STATUS (SS$_OVRDSKQUOTA);
;  308          1297  3          WRITE_QUOTA (.Q_RECORD);
;  309          1298  3          KERNEL_CALL (RET_QENTRY, .Q_RECORD, .ABD);
;  310          1299  2          END;
;  311          1300  2
;  312          1301  2      [FIB$C_ADD_QUOTA]:                  ! add quota file entry
;  313          1302  3          BEGIN
;  314          1303  3          IF .Q_RECORD NEQ 0
;  315          1304  3          THEN ERR_EXIT (SS$_DUPDSKQUOTA);
;  316          1305  3          IF .FREE_QUOTA EQL 0
;  317          1306  3          THEN
;  318          1307  4              BEGIN
;  319          1308  4              IF .FCB[FCB$L_FILESIZE] GEQU (1^24)/RECS_PER_BLOCK-1
;  320          1309  4              THEN ERR_EXIT (SS$_DEVICEFULL);
;  321          1310  4              TEMP1 = .FIB[FIB$W_CNTRLFUNC];
;  322          1311  4              TEMP2 = .FIB[FIB$L_CNTRLVAL];
;  323          1312  4              Q_RECORD = EXTEND_CONTIG (.FIB, .FCB, 1);
;  324          1313  4              MAKE_FCB_STALE (.FCB);
;  325          1314  4              FIB[FIB$W_CNTRLFUNC] = .TEMP1;
;  32           1315  4              FIB[FIB$L_CNTRLVAL] = .TEMP2;
;  3.7          1316  4              FIB[FIB$L_EXVBN] = 0;
;  328          1317  4              END
;  329          1318  3          ELSE
;  330          1319  4              BEGIN
;  331          1320  4              Q_RECORD = READ_BLOCK ((.FREE_QUOTA-1)/RECS_PER_BLOCK + .FCB[FCB$L_STLBN],
;  332          1321  4                                     1, QUOTA_TYPE);
;  333          1322  4              Q_RECORD = .Q_RECORD + ((.FREE_QUOTA-1) MOD RECS_PER_BLOCK) * DQF$C_LENGTH;
;  334          1323  3              END;
;  335          1324  3
;  336          1325  3          CH$FILL (0, DQF$C_LENGTH, .Q_RECORD);
;  337          1326  3          Q_RECORD[DQF$V_ACTIVE] = 1;
;  338          1327  3          Q_RECORD[DQF$L_UIC] = .Q_BLOCK[DQF$L_UIC];
;  339          1328  3          Q_RECORD[DQF$L_USAGE] = .Q_BLOCK[DQF$L_USAGE];
;  340          1329  3          Q_RECORD[DQF$L_PERMQUOTA] = .Q_BLOCK[DQF$L_PERMQUOTA];
;  341          1330  3          Q_RECORD[DQF$L_OVERDRAFT] = .Q_BLOCK[DQF$L_OVERDRAFT];
;  342          1331  3          WRITE_QUOTA (.Q_RECORD);
;  343          1332  2          END;
```

```
;  344          1333  2
;  345          1334  2     [INRANGE, OUTRANGE]:        0;      ! should not be called with other functions
;  346          1335  2
;  347          1336  2     TES;
;  348          1337  2
;  349          1338  1 END;                                    ! end of routine QUOTA_FILE_OP

                                                        .TITLE  QUOTAUTIL
                                                        .IDENT  \V04-001\

                                                        .EXTRN  MAKE_FCB_STALE, SERIAL_FILE
                                                        .EXTRN  ALLOCATION_LOCK
                                                        .EXTRN  SWITCH_VOLUME, SEARCH_QUOTA
                                                        .EXTRN  CHECK_PROTECT, GET_QUOTA_LOCK
                                                        .EXTRN  REL_QUOTA_LOCK, WRITE_DIRTY
                                                        .EXTRN  READ_BLOCK, EXTEND_CONTIG
                                                        .EXTRN  WRITE_QUOTA

                                                        .PSECT  $CODE$,NOWRT,2

                                  03FC 00000            .ENTRY  QUOTA_FILE_OP, Save R2,R3,R4,R5,R6,R7,R8,R9 ; 1105
                          59  0000G CF   9E 00002       MOVAB   WRITE_QUOTA, R9
                                     01   DD 00007       PUSHL   #1                                          ; 1196
                      0000G CF       01   FB 00009       CALLS   #1, SWITCH_VOLUME
                          50    98   AA   D0 0000E       MOVL    -104(BASE), R0                              ; 1197
                          58    54   A0   D0 00012       MOVL    84(R0), FCB
                        08 AA          58   D0 00016       MOVL    FCB, 8(BASE)
                                       05   12 0001A       BNEQ    1$                                          ; 1198
                                03D4   8F   BF 0001C       CHMU    #980                                        ; 1199
                                       04   00020         RET                                                 ; 1201
                            24   A8   9F   00021 1$:      PUSHAB  36(FCB)
                      0000G CF         01   FB 00024       CALLS   #1, SERIAL_FILE
                      0000G CF         00   FB 00029       CALLS   #0, ALLOCATION_LOCK                         ; 1203
                          50    08   AC   D0 0002E       MOVL    FIB, R0                                     ; 1210
                          0A    16   A0   B1 00032       CMPW    22(R0), #10
                                       03   12 00036       BNEQ    2$
                                  008F 31   00038         BRW     10$
                          50    04   AC   D0 0003B 2$:    MOVL    ABD, R0                                     ; 1213
                          20    12   A0   B1 0003F       CMPW    18(R0), #32
                                       05   1E 00043       BGEQU   3$
                                0114   8F   BF 00045       CHMU    #276                                        ; 1214
                                       04   00049         RET
                          51    04   AC   D0 0004A 3$:    MOVL    ABD, R1                                     ; 1215
                          50    10   A1   3C 0004E       MOVZWL  16(R1), R0
                          57    11 A140   9E 00052       MOVAB   17(R1)[R0], Q_BLOCK
                                     7E   D4 00057       CLRL    -(SP)                                       ; 1217
                          50    08   AC   D0 00059       MOVL    FIB, R0
                                10   A0   DD 0005D       PUSHL   16(R0)
                                18   A0   DD 00060       PUSHL   24(R0)
                                04   A7   DD 00063       PUSHL   4(Q_BLOCK)
                      0000G CF         04   FB 00066       CALLS   #4, SEARCH_QUOTA
                          56           50   D0 0006B       MOVL    R0, Q_RECORD
                          50    08   AC   D0 0006E       MOVL    FIB, R0                                     ; 1218
                          05    18   A0   E8 00072       BLBS    24(R0), 4$
                    06    18    A0         01   E1 00076       BBC     #1, 24(R0), 5$                          ; 1219
                          10    A0   02B4   CA   D0 0007B 4$:    MOVL    692(BASE), 16(R0)                       ; 1220
```

```
                        52      08   AC  D0 00081 5$:     MOVL    FIB, R2                                 ; 1227
                        0C      16   A2  B1 00085         CMPW    22(R2), #12                             ;
                                     09  13 00089         BEQL    6$                                      ;
                                     7E  D4 0008B         CLRL    -(SP)                                   ; 1229
                                     58  DD 0008D         PUSHL   FCB                                     ;
                        7E           01  7D 0008F         MOVQ    #1, -(SP)                               ;
                                     1E  11 00092         BRB     8$                                      ;
                        14      18   A2  E8 00094 6$:     BLBS    24(R2), 7$                              ; 1232
                0F      18   A2      01  E0 00098         BBS     #1, 24(R2), 7$                          ; 1233
                        50      90   AA  D0 0009D         MOVL    -112(BASE), R0                          ; 1235
                        50      58   A0  D0 000A1         MOVL    88(R0), R0                              ;
                             38 A0   A7  D1 000A5         CMPL    4(Q_BLOCK), 56(R0)                      ;
                                     0B  13 000AA         BEQL    9$                                      ;
                                     7E  D4 000AC 7$:     CLRL    -(SP)                                   ; 1236
                                     58  DD 000AE         PUSHL   FCB                                     ;
                                     7E  7C 000B0         CLRQ    -(SP)                                   ;
                        0000G        CF  04  FB 000B2 8$:     CALLS   #4, CHECK_PROTECT                   ;
                        50      08   AC  D0 000B7 9$:     MOVL    FIB, R0                                 ; 1243
                        0B      16   A0  B1 000BB         CMPW    22(R0), #11                             ;
                                     09  13 000BF         BEQL    10$                                     ;
                                     56  D5 000C1         TSTL    Q_RECORD                                ; 1245
                                     05  12 000C3         BNEQ    10$                                     ;
                             03E4    8F  BF 000C5         CHMU    #996                                    ; 1246
                                     04  000C9         RET                                             ;
                        50      08   AC  D0 000CA 10$:     MOVL    FIB, R0                                 ; 1252
                04      0A      16   A0  AF 000CE         CASEW   22(R0), #10, #4                         ;
        005E    00A5         00B0        000B     000D3 11$:     .WORD   12$-11$,-                       ;
                                         0025     000DB          23$-11$,-                               ;
                                                                 22$-11$,-                               ;
                                                                 16$-11$,-                               ;
                                                                 14$-11$                                 ;
                                     04  000DD         RET                                             ;
                        03      01   AA  E8 000DE 12$:     BLBS    1(BASE), 13$                            ; 1257
                                     24  BF 000E2         CHMU    #36                                     ; 1258
                                     04  000E4         RET                                             ;
                        0000V        CF  00  FB 000E5 13$:     CALLS   #0, FLUSH_QUO_CACHE                 ; 1259
                        7E           01  CE 000EA         MNEGL   #1, -(SP)                               ; 1260
                        0000G        CF  01  FB 000ED         CALLS   #1, WRITE_DIRTY                     ;
                        0000V        CF  00  FB 000F2         CALLS   #0, DEACC_QFILE                     ; 1261
                                     04  000F7         RET                                             ; 1252
                                 08  A6  D5 000F8 14$:     TSTL    8(Q_RECORD)                             ; 1271
                                     0A  13 000FB         BEQL    15$                                     ;
                        06       80  AA  E9 000FD         BLBC    -128(BASE), 15$                         ; 1272
                     80 AA       0669 8F  B0 00101         MOVW    #1641, -128(BASE)                       ;
                                 04  AC  DD 00107 15$:     PUSHL   ABD                                     ; 1273
                                     56  DD 0010A         PUSHL   Q_RECORD                                ;
                        0000V        CF  02  FB 0010C         CALLS   #2, RET_QENTRY                      ;
                                     05  DD 00111         PUSHL   #5                                      ; 1274
                                 02C0    CA  DD 00113         PUSHL   704(BASE)                           ;
                        0000G        CF  02  FB 00117         CALLS   #2, GET_QUOTA_LOCK                  ;
20          00           6E          00  2C 0011C         MOVC5   #0, (SP), #0, #32, (Q_RECORD)           ; 1275
                                         66      00121                                                   ;
                                     56  DD 00122         PUSHL   Q_RECORD                                ; 1276
                        69           01  FB 00124         CALLS   #1, WRITE_QUOTA                     ;
                                 02C0    CA  DD 00127         PUSHL   704(BASE)                           ; 1277
                        0000G        CF  01  FB 0012B         CALLS   #1, REL_QUOTA_LOCK                  ;
                                     04  00130         RET                                             ; 1252
```

```
                10      18      A0              02  E1 00131 16$:   BBC     #2, 24(R0), 18$                         ; 1282
                                        FF7C    CA  D5 00136         TSTL    -132(BASE)                              ; 1285
                                        05      12  00139A           BNEQ    17$                                     ;
                                        0800    8F  BF 0013C         CHMU    #2048                                   ; 1286
                                                    04 00140         RET                                             ;
                        08      A6      08      A7  D0 00141 17$:   MOVL    8(Q_BLOCK), 8(Q_RECORD)                 ; 1287
                        50      A6      08      AC  D0 00146 18$:   MOVL    FIB, R0                                 ; 1289
                05      18      A0              03  E1 0014A         BBC     #3, 24(R0), 19$                         ;
                        0C      A6      0C      A7  D0 0014F         MOVL    12(Q_BLOCK), 12(Q_RECORD)               ; 1291
                        50      A6      08      AC  D0 00154 19$:   MOVL    FIB, R0                                 ; 1292
                05      18      A0              04  E1 00158         BBC     #4, 24(R0), 20$                         ;
                        10      A6      10      A7  D0 0015D         MOVL    16(Q_BLOCK), 16(Q_RECORD)               ; 1294
                        0C      A6      08      A6  D1 00162 20$:   CMPL    8(Q_RECORD), 12(Q_RECORD)               ; 1295
                                                0A  1B 00167         BLEQU   21$                                     ;
                                06      80      AA  E9 00169         BLBC    -128(BASE), 21$                         ; 1296
                                80      AA      C669    8F  B0 00160         MOVW    #1641, -128(BASE)               ;
                                                56      DD 00173 21$:   PUSHL   Q_RECORD                            ; 1297
                                        69              01  FB 00175         CALLS   #1, WRITE_QUOTA                 ;
                                                04      AC  DD 00178 22$:   PUSHL   ABD                             ; 1298
                                                56      DD 0017B         PUSHL   Q_RECORD                            ;
                                0000V   CF              02  FB 0017D         CALLS   #2, RET_QENTRY                  ;
                                                            04 00182         RET                                     ; 1252
                                                56      D5 00183 23$:   TSTL    Q_RECORD                            ; 1303
                                                        05  13 00185         BEQL    24$                             ;
                                                03DC    8F  BF 00187         CHMU    #988                            ; 1304
                                                            04 0018B         RET                                     ;
                                        50      02B8    CA  D0 0018C 24$:   MOVL    696(BASE), R0                   ; 1305
                                                        49  12 00191         BNEQ    26$                             ;
                            000FFFFF    8F      38      A8  D1 00193         CMPL    56(FCB), #1048575               ; 1308
                                                        05  1F 0019B         BLSSU   25$                             ;
                                                0850    8F  BF 0019D         CHMU    #2128                           ; 1309
                                                            04 001A1         RET                                     ;
                                        50      08      AC  D0 001A2 25$:   MOVL    FIB, R0                         ; 1310
                                        53      16      A0  3C 001A6         MOVZWL  22(R0), TEMP1                   ;
                                        52      18      A0  D0 001AA         MOVL    24(R0), TEMP2                   ; 1311
                                                        01  DD 001AE         PUSHL   #1                              ; 1312
                                                0101    8F  BB 001B0         PUSHR   #^M<R0,R8>                      ;
                                0000G   CF              03  FB 001B4         CALLS   #3, EXTEND_CONTIG               ;
                                        56              50  D0 001B9         MOVL    R0, Q_RECORD                    ;
                                                        58  DD 001BC         PUSHL   FCB                             ; 1313
                                0000G   CF              01  FB 001BE         CALLS   #1, MAKE_FCB_STALE              ;
                                        50      08      AC  D0 001C3         MOVL    FIB, R0                         ; 1314
                                  16    A0              53  B0 001C7         MOVW    TEMP1, 22(R0)                   ;
                                        50      08      AC  D0 001CB         MOVL    FIB, R0                         ; 1315
                                  18    A0              52  D0 001CF         MOVL    TEMP2, 24(R0)                   ;
                                        50      08      AC  D0 001D3         MOVL    FIB, R0                         ; 1316
                                                1C      A0  D4 001D7         CLRL    28(R0)                          ;
                                                            2B 11 001DA         BRB     27$                             ; 1305
                                                            05 DD 001DC 26$:   PUSHL   #5                              ; 1320
                                                            01 DD 001DE         PUSHL   #1                              ;
                                                            50 D7 001E0         DECL    R0                              ;
                                        50                  10 C6 001E2         DIVL2   #16, R0                         ;
                                                30  B840    9F 001E5         PUSHAB  @48(FCB)[R0]                    ;
                                0000G   CF              03  FB 001E9         CALLS   #3, READ_BLOCK                  ;
                                        56              50  D0 001EE         MOVL    R0, Q_RECORD                    ;
                7E  FFFFFFFF    8F      02B8    CA          01  7A 001F1         EMUL    #1, 696(BASE), #-1, -(SP)       ; 1322
                50              50              8E          10  7B 001FC         EDIV    #16, (SP)+, R0, R0              ;
```

```
                        50              20  C4 00201        MULL2   #32, R0                                 ;
                        56              50  C0 00204        ADDL2   R0, Q_RECORD                            ;
        20      00              6E      00  2C 00207 27$:   MOVC5   #0, (SP), #0, #32, (Q_RECORD)          ; 1325
                                        66      0020C                                                       ;
                        66              01  88 0020D        BISB2   #1, (Q_RECORD)                          ; 1326
                04      A6      04      A7  7D 00210        MOVQ    4(Q_BLOCK), 4(Q_RECORD)                 ; 1327
                0C      A6      0C      A7  7D 00215        MOVQ    12(Q_BLOCK), 12(Q_RECORD)               ; 1329
                                        56  DD 0021A        PUSHL   Q_RECORD                                ; 1331
                        69              01  FB 0021C        CALLS   #1, WRITE_QUOTA                         ;
                                            04 0021F        RET                                             ; 1338

; Routine Size:  544 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  351     1339  1 GLOBAL ROUTINE FLUSH_QUO_CACHE : L_NORM NOVALUE =
;  352     1340  1
;  353     1341  1 !++
;  354     1342  1 !
;  355     1343  1 ! FUNCTIONAL DESCRIPTION:
;  356     1344  1 !
;  357     1345  1 !       This routine flushes dirty entries in the quota cache back to the
;  358     1346  1 !       quota file.
;  359     1347  1 !
;  360     1348  1 !
;  361     1349  1 ! CALLING SEQUENCE:
;  362     1350  1 !       FLUSH_QUO_CACHE ()
;  363     1351  1 !
;  364     1352  1 ! INPUT PARAMETERS:
;  365     1353  1 !       NONE
;  366     1354  1 !
;  367     1355  1 ! IMPLICIT INPUTS:
;  368     1356  1 !       CURRENT_VCB: VCB of volume
;  369     1357  1 !       context set to RVN 1
;  370     1358  1 !
;  371     1359  1 ! OUTPUT PARAMETERS:
;  372     1360  1 !       NONE
;  373     1361  1 !
;  374     1362  1 ! IMPLICIT OUTPUTS:
;  375     1363  1 !       NONE
;  376     1364  1 !
;  377     1365  1 ! ROUTINE VALUE:
;  378     1366  1 !       NONE
;  379     1367  1 !
;  380     1368  1 ! SIDE EFFECTS:
;  381     1369  1 !       quota cache flushed, quota file modified
;  382     1370  1 !
;  383     1371  1 !--
;  384     1372  1
;  385     1373  2 BEGIN
;  386     1374  2
;  387     1375  2 BUILTIN
;  388     1376  2         FP;
;  389     1377  2
;  390     1378  2 LITERAL
;  391     1379  2         RECS_PER_BLOCK  = 512 / DQF$C_LENGTH;
;  392     1380  2
;  393     1381  2 LOCAL
;  394     1382  2         QUOTA_CACHE     : REF BBLOCK,   ! address of quota cache
;  395     1383  2         QUOTA_LIST      : REF BBLOCKVECTOR [,VCA$C_QUOLENGTH],
;  396     1384  2                                         ! address of cache entries
;  397     1385  2         FCB             : REF BBLOCK,   ! address of quota file FCB
;  398     1386  2         REC_NUM,                        ! record number to read
;  399     1387  2         STATUS,                         ! system service status
;  400     1388  2         Q_RECORD        : REF BBLOCK,   ! address of record read
;  401     1389  2         LOCK_STATUS     : VECTOR [2];   ! LKSB for lock conversion
;  402     1390  2
;  403     1391  2
;  404     1392  2 BIND_COMMON;
;  405     1393  2
;  406     1394  2 EXTERNAL ROUTINE
;  407     1395  2         ZERO_ON_ERROR,                  ! return zero on error signal (handler)
```

```
;  408          1396  2          ALLOCATION_LOCK : L_NORM NOVALUE, ! serialize on volume
;  409          1397  2          READ_BLOCK      : L_NORM,        ! read a disk block
;  410          1398  2          CLEAR_QUO_CACHE : L_NORM,        ! flush cache entry to record
;  411          1399  2          REL_QUOTA_LOCK  : L_NORM;        ! release lock on cache entry
;  412          1400  2
;  413          1401  2
;  414          1402  2  ! Set up the condition handler to handle I/O errors.
;  415          1403  2  !
;  416          1404  2
;  417          1405  2  .FP = ZERO_ON_ERROR;
;  418          1406  2
;  419          1407  2  ! Scan the quota cache, looking for valid dirty entries. If one is found,
;  420          1408  2  ! read its record from the quota file, update the record, and write it back.
;  421          1409  2  !
;  422          1410  2
;  423          1411  2  QUOTA_CACHE = .CURRENT_VCB[VCB$L_QUOCACHE];
;  424          1412  2  IF .QUOTA_CACHE EQL 0 THEN RETURN;       ! nop if no quota cache
;  425          1413  2
;  426          1414  2  ALLOCATION_LOCK ();
;  427          1415  2
;  428          1416  2  FCB = .CURRENT_VCB[VCB$L_QUOTAFCB];
;  429          1417  2  QUOTA_LIST = QUOTA_CACHE[VCA$L_QUOLIST];
;  430          1418  2  INCR J FROM 1 TO .QUOTA_CACHE[VCA$W_QUOSIZE]
;  431          1419  2  DO
;  432          1420  3      BEGIN
;  433          1421  3      IF  .QUOTA_LIST[.J-1, VCA$V_QUODIRTY]
;  434          1422  3      AND .QUOTA_LIST[.J-1, VCA$L_QUORECNUM] NEQ 0
;  435          1423  3      THEN
;  436          1424  4          BEGIN
;  437          1425  4          REC_NUM = .QUOTA_LIST[.J-1, VCA$L_QUORECNUM] - 1;
;  438          1426  4          Q_RECORD = READ_BLOCK (.REC_NUM / RECS_PER_BLOCK
;  439          1427  4                                + .FCB[FCB$L_ST[BN], 1, QUOTA_TYPE)
;  440          1428  4                    + (.REC_NUM MOD RECS_PER_BLOCK) * DQF$C_LENGTH;
;  441          1429  4          IF .Q_RECORD GEQA 512
;  442          1430  4          THEN KERNEL_CALL (CLEAN_QUO_CACHE, .J, .Q_RECORD);
;  443          1431  3          END;
;  444          1432  3      REL_QUOTA_LOCK (.J);
;  445          1433  2      END;
;  446          1434  2
;  447          1435  2  ! Now mark the quota cache invalid. If we are holding a cache lock,
;  448          1436  2  ! demote it down to NL to indicate that we are no longer holding
;  449          1437  2  ! cache contents.
;  450          1438  2  !
;  451          1439  2
;  452          1440  2  QUOTA_CACHE[VCA$V_CACHEVALID] = 0;
;  453          1441  2  IF .QUOTA_CACHE[VCA$L_QUOCLKID] NEQ 0
;  454          1442  2  THEN
;  455          1443  3      BEGIN
;  456          1444  3      LOCK_STATUS[1] = .QUOTA_CACHE[VCA$L_QUOCLKID];
;  457        P 1445  3      STATUS = $ENQW (EFN    = EFN,
;  458        P 1446  3                      LKMODE = LCK$K_NLMODE,
;  459        P 1447  3                      FLAGS  = LCK$M_NOQUEUE OR LCK$M_SYNCSTS OR LCK$M_CVTSYS OR LCK$M_CONVERT,
;  460        P 1448  3                      LKSB   = LOCK_STATUS
;  461          1449  3                      );
;  462          1450  3      IF NOT .STATUS
;  463          1451  3      THEN BUG_CHECK (XQPERR, FATAL, 'Unexpected lock manager error');
;  464          1452  2      END;
```

```
;  465          1453  2
;  466          1454  1 END;                                  ! end of routine FLUSH_QUO_CACHE


                                                              .EXTRN  ZERO_ON_ERROR, CLEAN_QUO_CACHE
                                                              .EXTRN  SYS$ENQW, BUG$_XQPERR

                                                01FC 00000    .ENTRY  FLUSH_QUO_CACHE, Save R2,R3,R4,R5,R6,R7,R8 ; 1339
                                             5E             08  C2 00002    SUBL2   #8, SP
                                             6D      0000G  CF  9E 00005    MOVAB   ZERO_ON_ERROR, (FP)                ; 1405
                                             50        98   AA  D0 0000A    MOVL    -104(BASE), R0                     ; 1411
                                             52        5C   A0  D0 0000E    MOVL    92(R0), QUOTA_CACHE
                                                            7A  13 00012    BEQL    4$                                 ; 1412
                                          0000G  CF         00  FB 00014    CALLS   #0, ALLOCATION_LOCK                ; 1414
                                             50        98   AA  D0 00019    MOVL    -104(BASE), R0                     ; 1416
                                             56        54   A0  D0 0001D    MOVL    84(R0), FCB
                                             53        44   A2  9E 00021    MOVAB   68(R2), QUOTA_LIST                 ; 1417
                                             58             62  3C 00025    MOVZWL  (QUOTA_CACHE), R8                  ; 1418
                                                            54  D4 00028    CLRL    J
                                                            57  11 0002A    BRB     3$
                                                50          54  1C  C5 0002C 1$:    MULL3   #28, J, R0                 ; 1421
                                                            50  53  C0 00030    ADDL2   QUOTA_LIST, R0
                                                        44  EF  A0  01  E1 00033    BBC     #1, -T7(R0), 2$
                                          00            EC  A0      18  00  ED 00038    CMPZV   #0, #24, -20(R0), #0       ; 1422
                                                                    3C  13 0003E    BEQL    2$
                                          55            EC  A0      18  00  EF 00040    EXTZV   #0, #24, -20(R0), REC_NUM  ; 1425
                                                                    55  D7 00046    DECL    REC_NUM
                                                                    05  DD 00048    PUSHL   #5                         ; 1426
                                                                    01  DD 0004A    PUSHL   #1
                                                50                  55      10  C7 0004C    DIVL3   #16, REC_NUM, R0
                                                                 30 B640    9F 00050    PUSHAB  @48(FCB)[R0]           ; 1427
                                                      0000G  CF         03  FB 00054    CALLS   #3, READ_BLOCK
                                          7E            00            55      01  7A 00059    EMUL    #1, REC_NUM, #0, -(SP) ; 1428
                                          51            51            8E      10  7B 0005E    EDIV    #16, (SP)+, R1, R1
                                                                      51      20  C4 00063    MULL2   #32, R1
                                                      57              50      51  C1 00066    ADDL3   R1, R0, Q_RECORD
                                                   00000200   8F              57  D1 0006A    CMPL    Q_RECORD, #512     ; 1429
                                                                               09  1F 00071    BLSSU   2$
                                                                        0090   8F  BB 00073    PUSHR   #^M<R4,R7>         ; 1430
                                                      0000G  CF         02  FB 00077    CALLS   #2, CLEAN_QUO_CACHE
                                                                              54  DD 0007C 2$:    PUSHL   J              ; 1432
                                                      0000G  CF         01  FB 0007E    CALLS   #1, REL_QUOTA_LOCK
                                                   A5                 54      58  F3 00083 3$:    AOBLEQ  R8, J, 1$      ; 1418
                                                         0B   A2          01  8A 00087    BICB2   #1, 11(QUOTA_CACHE)    ; 1440
                                                                      04   A2  D5 0008B    TSTL    4(QUOTA_CACHE)         ; 1441
                                                                              25  13 0008E 4$:    BEQL    5$
                                                         04   AE       04   A2  D0 00090    MOVL    4(QUOTA_CACHE), LOCK_STATUS+4 ; 1444
                                                                              7E  7C 00095    CLRQ    -(SP)              ; 1449
                                                                              7E  7C 00097    CLRQ    -(SP)
                                                                              7E  7C 00099    CLRQ    -(SP)
                                                                              7E  D4 0009B    CLRL    -(SP)
                                                                  7E      4E  8F  9A 0009D    MOVZBL  #78, -(SP)
                                                                       20 AE  9F  9F 000A1    PUSHAB  LOCK_STATUS
                                                                  7E          1E  7D 000A4    MOVQ    #30, -(SP)
                                                   00000000G      00          0B  FB 000A7    CALLS   #11, SYS$ENQW
                                                                  04          50  E8 000AF    BLBS    STATUS, 5$         ; 1450
```

```
                FEFF 000B1          BUGW                                        ; 1451
              0000* 000B3           .WORD   <BUG$_XQPERR!4>                     ;
                 04 000B5 5$:       RET                                         ; 1454

; Routine Size:  182 bytes,    Routine Base:  $CODE$ + 0220
```

```
;  468     1455  1 GLOBAL ROUTINE DEACC_QFILE : L_NORM =
;  469     1456  1
;  470     1457  1 !++
;  471     1458  1 !
;  472     1459  1 ! FUNCTIONAL DESCRIPTION:
;  473     1460  1 !
;  474     1461  1 !       This routine deaccesses the quota file and releases the FCB if it
;  475     1462  1 !       is idle. This routine must be aclled in kernel mode.
;  476     1463  1 !
;  477     1464  1 ! CALLING SEQUENCE:
;  478     1465  1 !       DEACC_QFILE ()
;  479     1466  1 !
;  480     1467  1 ! INPUT PARAMETERS:
;  481     1468  1 !       NONE
;  482     1469  1 !
;  483     1470  1 ! IMPLICIT INPUTS:
;  484     1471  1 !       CURRENT_VCB: VCB of volume
;  485     1472  1 !       context set to RVN 1
;  486     1473  1 !
;  487     1474  1 ! OUTPUT PARAMETERS:
;  488     1475  1 !       NONE
;  489     1476  1 !
;  490     1477  1 ! IMPLICIT OUTPUTS:
;  491     1478  1 !       NONE
;  492     1479  1 !
;  493     1480  1 ! ROUTINE VALUE:
;  494     1481  1 !       1
;  495     1482  1 !
;  496     1483  1 ! SIDE EFFECTS:
;  497     1484  1 !       quota file disconnected from VCB, FCB deallocated
;  498     1485  1 !
;  499     1486  1 !--
;  500     1487  1
;  501     1488  2 BEGIN
;  502     1489  2
;  503     1490  2 LOCAL
;  504     1491  2         ACCTL,                          ! calculate remaining access control
;  505     1492  2         LCKMODE,                        ! lock mode to convert access lock to.
;  506     1493  2         FCB             : REF BBLOCK,   ! FCB of quota file
;  507     1494  2         STATUS,                         ! system service status
;  508     1495  2         QUOTA_CACHE     : REF BBLOCK;   ! address of quota cache block
;  509     1496  2
;  510     1497  2 BIND_COMMON;
;  511     1498  2
;  512     1499  2 EXTERNAL ROUTINE
;  513     1500  2         KILL_BUFFERS    : L_NORM,       ! flush specified buffers from cache
;  514     1501  2         CONV_ACCLOCK    : L_NORM,       ! convert access lock
;  515     1502  2         LOCK_MODE       : L_JSB_1ARG,   ! calculate lock mode from access ctl
;  516     1503  2         DEQ_LOCK        : L_NORM,       ! dequeue a lock
;  517     1504  2         DEALLOCATE      : L_NORM ADDRESSING_MODE (GENERAL); ! deallocate system dynamic memory
;  518     1505  2
;  519     1506  2
;  520     1507  2 ! Flush the quota file data blocks from the block buffer cache.
;  521     1508  2 !
;  522     1509  2
;  523     1510  2 KILL_BUFFERS (1, -1);
;  524     1511  2
```

```
;  525          1512  2 ! Decrement access and lock counts on the FCB.
;  526          1513  2 !
;  527          1514  2
;  528          1515  2 PRIMARY_FCB = FCB = .CURRENT_VCB[VCB$L_QUOTAFCB];
;  529          1516  2 CURRENT_VCB[VCB$L_QUOTAFCB] = 0;
;  530          1517  2
;  531          1518  2 ACCTL = 0;
;  532          1519  2
;  533          1520  2 IF .FCB[FCB$W_WCNT] NEQ 0
;  534          1521  2 THEN ACCTL = FIB$M_WRITE;
;  535          1522  2
;  536          1523  2 FCB[FCB$W_TCNT] = .FCB[FCB$W_TCNT] - 1;
;  537          1524  2
;  538          1525  2 LCKMODE = 0;
;  539          1526  2
;  540          1527  2 IF (FCB[FCB$W_ACNT] = .FCB[FCB$W_ACNT] - 1) NEQ 0
;  541          1528  2 THEN
;  542          1529  2     LCKMODE = LOCK_MODE (.ACCTL);
;  543          1530  2
;  544          1531  2 FCB[FCB$W_REFCNT] = .FCB[FCB$W_REFCNT] - 1;
;  545          1532  2
;  546          1533  2 ! Convert the access lock to reflect the remaining accessors.
;  547          1534  2 !
;  548          1535  2
;  549          1536  2 CONV_ACCLOCK (.LCKMODE, .FCB);
;  550          1537  2
;  551          1538  2 ! Release the quota cache lock, if there was one. Unlink and deallocate
;  552          1539  2 ! the quota cache block.
;  553          1540  2 !
;  554          1541  2
;  555          1542  2 QUOTA_CACHE = .CURRENT_VCB[VCB$L_QUOCACHE];
;  556          1543  2 IF .QUOTA_CACHE[VCA$L_QUOCLKID] NEQ 0
;  557          1544  2 THEN
;  558          1545  3     BEGIN
;  559          1546  3     DEQ_LOCK (.QUOTA_CACHE[VCA$L_QUOCLKID]);
;  560          1547  2     END;
;  561          1548  2
;  562          1549  2 DEALLOCATE (.QUOTA_CACHE);
;  563          1550  2 CURRENT_VCB[VCB$L_QUOCACHE] = 0;
;  564          1551  2
;  565          1552  2 RETURN 1;
;  566          1553  2
;  567          1554  1 END;                                   ! end of routine DEACC_QFILE


                                                .FXTRN  KILL_BUFFERS, CONV_ACCLOCK
                                                .EXTRN  LOCK_MODE, DEQ_LOCK
                                                .EXTRN  DEALLOCATE

                                   000C 00000   .ENTRY  DEACC_QFILE, Save R2,R3                ; 1455
                         7E            01  CE  00002   MNEGL   #1, -(SP)                      ; 1510
                                       01  DD  00005   PUSHL   #1                             ;
                   0000G CF            02  FB  00007   CALLS   #2, KILL_BUFFERS              ;
                         50      98    AA  D0  0000C   MOVL    -104(BASE), R0                 ; 1515
                         52      54    A0  D0  00010   MOVL    84(R0), FCB                    ;
                      08 AA            52  D0  00014   MOVL    FCB, 8(BASE)                   ;
```

```
                50      98    AA   D0 00018          MOVL    -104(BASE), R0                  ; 1516
                        54    A0   D4 0001C          CLRL    84(R0)
                             50   D4 0001F          CLRL    ACCTL                           ; 1518
                        1C    A2   B5 00021          TSTW    28(FCB)                         ; 1520
                             05   13 00024          BEQL    1$
                50      0100  8F   3C 00026          MOVZWL  #256, ACCTL                     ; 1521
                        20    A2   B7 0002B 1$:      DECW    32(FCB)                         ; 1523
                             53   D4 0002E          CLRL    LCKMODE                         ; 1525
                51      1A    A2   3C 00030          MOVZWL  26(FCB), R1                     ; 1527
                             51   D7 00034          DECL    R1
             1A A2              51   B0 00036          MOVW    R1, 26(FCB)
                             51   D5 0003A          TSTL    R1
                             06   13 0003C          BEQL    2$
                          0000G 30 0003E          BSBW    LOCK_MODE                       ; 1529
                53              50   D0 00041          MOVL    R0, LCKMODE
                        18    A2   B7 00044 2$:      DECW    24(FCB)                         ; 1531
                             52   DD 00047          PUSHL   FCB                             ; 1536
                             53   DD 00049          PUSHL   LCKMODE
          0000G CF              02   FB 0004B          CALLS   #2, CONV_ACCLOCK
                50      98    AA   D0 00050          MOVL    -104(BASE), R0                  ; 1542
                52      5C    A0   D0 00054          MOVL    92(R0), QUOTA_CACHE
                        04    A2   D5 00058          TSTL    4(QUOTA_CACHE)                  ; 1543
                             08   13 0005B          BEQL    3$
                        04    A2   DD 0005D          PUSHL   4(QUOTA_CACHE)                  ; 1546
          0000G CF              01   FB 00060          CALLS   #1, DEQ_LOCK
                             52   DD 00065 3$:      PUSHL   QUOTA_CACHE                     ; 1549
 00000000G 00                 01   FB 00067          CALLS   #1, DEALLOCATE
                50      98    AA   D0 0006E          MOVL    -104(BASE), R0                  ; 1550
                        5C    A0   D4 00072          CLRL    92(R0)
                50              01   D0 00075          MOVL    #1, R0                          ; 1552
                                  04   00078          RET                                     ; 1554

; Routine Size:  121 bytes,    Routine Base:  $CODE$ + 02D6
```

```
;  569     1555  1 GLOBAL ROUTINE RET_QENTRY (Q_RECORD, ABD) : L_NORM =
;  570     1556  1
;  571     1557  1 !++
;  572     1558  1 !
;  573     1559  1 ! FUNCTIONAL DESCRIPTION:
;  574     1560  1 !
;  575     1561  1 !       This routine copies the specified quota file record into the
;  576     1562  1 !       result string area of the buffer descriptor packet. This routine
;  577     1563  1 !       must be called in kernel mode.
;  578     1564  1 !
;  579     1565  1 ! CALLING SEQUENCE:
;  580     1566  1 !       RET_QENTRY (ARG1, ARG2)
;  581     1567  1 !
;  582     1568  1 ! INPUT PARAMETERS:
;  583     1569  1 !       ARG1: address of quota file record
;  584     1570  1 !
;  585     1571  1 ! IMPLICIT INPUTS:
;  586     1572  1 !       NONE
;  587     1573  1 !
;  588     1574  1 ! OUTPUT PARAMETERS:
;  589     1575  1 !       ARG2: address of buffer descriptor packet
;  590     1576  1 !
;  591     1577  1 ! IMPLICIT OUTPUTS:
;  592     1578  1 !       NONE
;  593     1579  1 !
;  594     1580  1 ! ROUTINE VALUE:
;  595     1581  1 !       1
;  596     1582  1 !
;  597     1583  1 ! SIDE EFFECTS:
;  598     1584  1 !       NONE
;  599     1585  1 !
;  600     1586  1 !--
;  601     1587  1
;  602     1588  2 BEGIN
;  603     1589  2
;  604     1590  2 MAP
;  605     1591  2         Q_RECORD        : REF BBLOCK,   ! quota file record
;  606     1592  2         ABD             : REF BBLOCKVECTOR [,ABD$C_LENGTH];
;  607     1593  2                                         ! descriptor arg
;  608     1594  2
;  609     1595  2 ! If the user provided a result length buffer, give him the length
;  610     1596  2 ! of the record.
;  611     1597  2 !
;  612     1598  2
;  613     1599  2 IF .ABD[ABD$C_RESL, ABD$W_COUNT] GEQ 2
;  614     1600  2 THEN
;  615     1601  3     BEGIN
;  616     1602  3     (.ABD[ABD$C_RESL, ABD$W_TEXT] + ABD[ABD$C_RESL, ABD$W_TEXT] + 1)<0,16> = DQF$C_LENGTH;
;  617     1603  2     END;
;  618     1604  2
;  619     1605  2 ! If the user provided a result string buffer, return as much of the
;  620     1606  2 ! quota record as will fit (zero filling the buffer).
;  621     1607  2 !
;  622     1608  2
;  623     1609  2 CH$COPY (DQF$C_LENGTH, .Q_RECORD, 0,
;  624     1610  2     .ABD[ABD$C_RES, ABD$W_COUNT],
;  625     1611  2     .ABD[ABD$C_RES, ABD$W_TEXT] + ABD[ABD$C_RES, ABD$W_TEXT] + 1);
```

```
;  626          1612  2
;  627          1613  2 RETURN 1;
;  628          1614  2
;  629          1615  1 END;                                   ! end of routine RET_QENTRY



                                                003C 00000          .ENTRY  RET_QENTRY, Save R2,R3,R4,R5                    ; 1555
                                        50      08   AC   D0 00002  MOVL    ABD, R0                                         ; 1599
                                        02      1A   A0   B1 00006  CMPW    26(R0), #2                                      ;
                                                     0E   1F 0000A  BLSSU   1$                                              ;
                                        51      18   A0   9E 0000C  MOVAB   24(R0), R1                                      ; 1602
                                        50           61   3C 00010  MOVZWL  (R1), R0                                       ;
                                                01 A140   9F 00013  PUSHAB  1(R1)[R0]                                       ;
                                        9E           20   B0 00017  MOVW    #32, @(SP)+                                     ;
                                        52      08   AC   D0 0001A 1$:  MOVL    ABD, R2                                     ; 1610
                                        51      20   A2   9E 0001E  MOVAB   32(R2), R1                                     ; 1611
                                        50           61   3C 00022  MOVZWL  (R1), R0                                       ;
      22   A2              00      04   BC           20   2C 00025  MOVC5   #32, @Q_RECORD, #0, 34(R2), 1(R1)[R0]          ;
                                                01 A140        0002C                                                        ;
                                        50           01   D0 0002F  MOVL    #1, R0                                          ; 1613
                                                          04 00032  RET                                                     ; 1615

; Routine Size:  51 bytes,    Routine Base:  $CODE$ + 034F
```

```
;  631     1616  1 GLOBAL ROUTINE CONN_QFILE (ABD, FIB) : L_NORM NOVALUE =
;  632     1617  1
;  633     1618  1 !++
;  634     1619  1 !
;  635     1620  1 ! FUNCTIONAL DESCRIPTION:
;  636     1621  1 !
;  637     1622  1 !       This routine causes the quota file for the volume set to be
;  638     1623  1 !       connected and made active.
;  639     1624  1 !
;  640     1625  1 ! CALLING SEQUENCE:
;  641     1626  1 !       CONN_QFILE (ARG1, ARG2)
;  642     1627  1 !
;  643     1628  1 ! INPUT PARAMETERS:
;  644     1629  1 !       ARG1: address of buffer descriptor vector
;  645     1630  1 !       ARG2: address of user FIB
;  646     1631  1 !
;  647     1632  1 ! IMPLICIT INPUTS:
;  648     1633  1 !       CLEANUP_FLAGS: cleanup action and status flags
;  649     1634  1 !       CURRENT_RVN: RVN of currently selected volume
;  650     1635  1 !       CURRENT_VCB: VCB of currently selected volume
;  651     1636  1 !
;  652     1637  1 ! OUTPUT PARAMETERS:
;  653     1638  1 !       NONE
;  654     1639  1 !
;  655     1640  1 ! IMPLICIT OUTPUTS:
;  656     1641  1 !       PRIMARY_FCB: FCB created for quota file
;  657     1642  1 !
;  658     1643  1 ! ROUTINE VALUE:
;  659     1644  1 !       NONE
;  660     1645  1 !
;  661     1646  1 ! SIDE EFFECTS:
;  662     1647  1 !       directory searched, quota file accessed (FCB created, etc.)
;  663     1648  1 !
;  664     1649  1 !--
;  665     1650  1
;  666     1651  2 BEGIN
;  667     1652  2
;  668     1653  2 MAP
;  669     1654  2         ABD             : REF BBLOCKVECTOR [,ABD$C_LENGTH],
;  670     1655  2                                         ! buffer descriptor arg
;  671     1656  2         FIB             : REF BBLOCK;   ! user FIB
;  672     1657  2
;  673     1658  2 LOCAL
;  674     1659  2         FCB             : REF BBLOCK,   ! FCB of quota file
;  675     1660  2         HEADER          : REF BBLOCK,   ! file header of quota file
;  676     1661  2         BUFFER          : REF BBLOCK;   ! disk block buffer
;  677     1662  2
;  678     1663  2 BIND_COMMON;
;  679     1664  2
;  680     1665  2 EXTERNAL ROUTINE
;  681     1666  2         REBLD_PRIM_FCB  : L_NORM NOVALUE, ! rebuild fcb from header
;  682     1667  2         BUILD_EXT_FCBS  : L_NORM NOVALUE, ! build extension fcbs
;  683     1668  2         ARBITRATE_ACCESS : [ JSB_2ARGS,   ! arbitrate file access
;  684     1669  2         SERIAL_FILE     : L_NORM,         ! serialize on given file
;  685     1670  2         FIND            : L_NORM,         ! find file in directory
;  686     1671  2         SWITCH_VOLUME   : L_NORM,         ! switch volume context
;  687     1672  2         SEARCH_FCB      : L_NORM ADDRESSING_MODE (GENERAL), ! search FCB list
```

```
;  688          1673  2          READ_HEADER     : L_NORM,       ! read file header
;  689          1674  2          CREATE_FCB      : L_NORM;       ! create an FCB
;  690          1675  2
;  691          1676  2
;  692          1677  2 ! Check caller privilege - must be "system".
;  693          1678  2 !
;  694          1679  2
;  695          1680  2 IF NOT .CLEANUP_FLAGS[CLF_SYSPRV]
;  696          1681  2 THEN ERR_EXIT (SS$_NOPRIV);
;  697          1682  2
;  698          1683  2 ! Find the quota file in the directory. The quota file must be located
;  699          1684  2 ! RVN 1 if this is a volume set.
;  700          1685  2 !
;  701          1686  2
;  702          1687  2 IF .CLEANUP_FLAGS[CLF_DIRECTORY]
;  703          1688  2 THEN FIND (.ABD, .FIB, 0);
;  704          1689  2 SWITCH_VOLUME (.FIB[FIB$W_FID_RVN]);
;  705          1690  2 IF .CURRENT_RVN GTRU 1
;  706          1691  2 THEN ERR_EXIT (SS$_BADQFILE);
;  707          1692  2
;  708          1693  2 ! Make sure the quota file is not already active.
;  709          1694  2 !
;  710          1695  2
;  711          1696  2 IF .CURRENT_VCB[VCB$L_QUOTAFCB] NEQ 0
;  712          1697  2 THEN ERR_EXIT (SS$_QFACTIVE);
;  713          1698  2
;  714          1699  2 ! Find the FCB, if any, and read the header.
;  715          1700  2 !
;  716          1701  2
;  717          1702  2 SERIAL_FILE (FIB [FIB$W_FID]);
;  718          1703  2
;  719          1704  2 FCB = PRIMARY_FCB = SEARCH_FCB (FIB[FIB$W_FID]);
;  720          1705  2
;  721          1706  2 HEADER = READ_HEADER (FIB[FIB$W_FID]);
;  722          1707  2
;  723          1708  2 ! Create an FCB if none exists
;  724          1709  2 !
;  725          1710  2
;  726          1711  2 IF .FCB EQL 0
;  727          1712  2 THEN
;  728          1713  2     PRIMARY_FCB = FCB = CREATE_FCB (.HEADER)
;  729          1714  2 ELSE
;  730          1715  2     IF .FCB [FCB$V_STALE]
;  731          1716  2     THEN
;  732          1717  2         REBLD_PRIM_FCB (.FCB, .HEADER);
;  733          1718  2
;  734           719  2 BUILD_EXT_FCBS (.HEADER);
;  735          1720  2
;  736          1721  2 ! Check the quota file for suitability (contiguous, file format, etc.)
;  737          1722  2 !
;  738          1723  2
;  739          1724  2 IF NOT .HEADER[FH2$V_CONTIG]
;  740          1725  2 OR .BBLOCK [HEADER[FH2$W_RECATTR], FAT$B_RTYPE] NEQ FAT$C_FIXED
;  741          1726  2 OR .BBLOCK [HEADER[FH2$W_RECATTR], FAT$B_RATTR]B] NEQ 0
;  742          1727  2 OR .BBLOCK [HEADER[FH2$W_RECATTR], FAT$W_RSIZE] NEQ DQF$C_LENGTH
;  743          1728  2 THEN ERR_EXIT (SS$_BADQFILE);
;  744          1729  2
```

```
;  745          1730  2 ! Check access interlocks.
;  746          1731  2 !
;  747          1732  2
;  748          1733  2 IF NOT ARBITRATE_ACCESS (0, .FCB)
;  749          1734  2 THEN ERR_EXIT (SS$_ACCONFLICT);
;  750          1735  2
;  751          1736  2 ! Now hook up the quota file FCB.
;  752          1737  2 !
;  753          1738  2
;  754          1739  2 IF NOT KERNEL_CALL (MAKE_QFCB, .FCB)
;  755          1740  2 THEN ERR_EXIT (SS$_INSFMEM);          ! allocation failure on quota cache
;  756          1741  2
;  757          1742  1 END;                                  ! end of routine CONN_QFILE


                                                        .EXTRN  REBLD_PRIM_FCB, BUILD_EXT_FCBS
                                                        .EXTRN  ARBITRATE_ACCESS
                                                        .EXTRN  FIND, SEARCH_FCB
                                                        .EXTRN  READ_HEADER, CREATE_FCB

                                        000C 00000      .ENTRY  CONN_QFILE, Save R2,R3                  ; 1616
                          03        01  AA  E8 00002      BLBS    1(BASE), 1$                         ; 1680
                                        24  BF 00006      CHMU    #36                                 ; 1681
                                            04 00008      RET                                         ;
                    08                  06  E1 00009 1$:  BBC     #6, (BASE), 2$                      ; 1687
                                        7E  D4 0000D      CLRL    -(SP)                               ; 1688
                          7E        04  AC  7D 0000F      MOVQ    ABD, -(SP)                          ;
                 0000G    CF            03  FB 00013      CALLS   #3, FIND                            ;
                          50        08  AC  D0 00018 2$:  MOVL    FIB, R0                             ; 1689
                          7E        08  A0  3C 0001C      MOVZWL  8(R0), -(SP)                        ;
                 0000G    CF            01  FB 00020      CALLS   #1, SWITCH_VOLUME                   ;
                          01        A0  AA  D1 00025      CMPL    -96(BASE), #1                       ; 1690
                                        76  1A 00029      BGTRU   6$                                  ;
                          50        98  AA  D0 0002B      MOVL    -104(BASE), R0                      ; 1696
                                    54  A0  D5 0002F      TSTL    84(R0)                              ;
                                        05  13 00032      BEQL    3$                                  ;
                                03CC    8F  BF 00034      CHMU    #972                                ; 1697
                                            04 00038      RET                                         ;
             7E       08        AC          04  C1 00039 3$:  ADDL3   #4, FIB, -(SP)                  ; 1702
                 0000G    CF            01  FB 0003E      CALLS   #1, SERIAL_FILE                     ;
             7E       08        AC          04  C1 00043      ADDL3   #4, FIB, -(SP)                  ; 1704
          00000000G   00            01  FB 00048      CALLS   #1, SEARCH_FCB                      ;
                   08   AA          50  D0 0004F      MOVL    R0, 8(BASE)                         ;
                        53          50  D0 00053      MOVL    R0, FCB                             ;
             7E       08        AC          04  C1 00056      ADDL3   #4, FIB, -(SP)                  ; 1706
                 0000G    CF            01  FB 0005B      CALLS   #1, READ_HEADER                     ;
                          52            50  D0 00060      MOVL    R0, HEADER                          ;
                                        53  D5 00063      TSTL    FCB                                 ; 1711
                                        10  12 00065      BNEQ    4$                                  ;
                                        52  DD 00067      PUSHL   HEADER                              ; 1713
                 0000G    CF            01  FB 00069      CALLS   #1, CREATE_FCB                      ;
                          53            50  D0 0006E      MOVL    R0, FCB                             ;
                   08   AA          53  D0 00071      MOVL    FCB, 8(BASE)                        ;
                                        0D  11 00075      BRB     5$                                  ;
                          09        23  A3  E9 00077 4$:  BLBC    35(FCB), 5$                         ; 1715
                                        52  DD 0007B      PUSHL   HEADER                              ; 1717
```

```
                                53  DD 0007D          PUSHL   FCB                                     ;
                      0000G     CF  02  FB 0007F          CALLS   #2, REBLD_PRIM_FCB                      ;
                                52  DD 00084 5$:        PUSHL   HEADER                                  ; 1719
                      0000G     CF  01  FB 00086          CALLS   #1, BUILD_EXT_FCBS                      ;
                            34      A2  95 0008B          TSTB    52(HEADER)                              ; 1724
                                      11  18 0008E          BGEQ    6$                                      ;
                            01    14  A2  91 00090          CMPB    20(HEADER), #1                          ; 1725
                                      0B  12 00094          BNEQ    6$                                      ;
                                15      A2  95 00096          TSTB    21(HEADER)                              ; 1726
                                      06  12 00099          BNEQ    6$                                      ;
                            20    16  A2  B1 0009B          CMPW    22(HEADER), #32                         ; 1727
                                      05  13 0009F          BEQL    7$                                      ;
                          03BC    8F  BF 000A1 6$:        CHMU    #956                                    ; 1728
                                      04  000A5          RET                                             ;
                            51        53  D0 000A6 7$:        MOVL    FCB, R1                                 ; 1733
                                      50  D4 000A9          CLRL    R0                                      ;
                                  0000G 30 000AB          BSBW    ARBITRATE_ACCESS                        ;
                            05        50  E8 000AE          BLBS    R0, 8$                                  ;
                          0800    8F  BF 000B1          CHMU    #2048                                   ; 1734
                                      04  000B5          RET                                             ;
                                53  DD 000B6 8$:        PUSHL   FCB                                     ; 1739
                      0000V     CF  01  FB 000B8          CALLS   #1, MAKE_QFCB                           ;
                            04        50  E8 000BD          BLBS    R0, 9$                                  ;
                          0124    8F  BF 000C0          CHMU    #292                                    ; 1740
                                      04  000C4 9$:        RET                                             ; 1742

; Routine Size:  197 bytes,    Routine Base:  $CODE$ + 0382
```

```
;  759          1743  1 GLOBAL ROUTINE MAKE_QFCB (FCB) : L_NORM =
;  760          1744  1
;  761          1745  1 !++
;  762          1746  1 !
;  763          1747  1 ! FUNCTIONAL DESCRIPTION:
;  764          1748  1 !
;  765          1749  1 !       This routine hooks up the specified FCB to be the FCB for the
;  766          1750  1 !       volume (set) quota file. This routine must be called in kernel mode.
;  767          1751  1 !
;  768          1752  1 ! CALLING SEQUENCE:
;  769          1753  1 !       MAKE_QFCB (ARG1)
;  770          1754  1 !
;  771          1755  1 ! INPUT PARAMETERS:
;  772          1756  1 !       ARG1: address of FCB to hook up
;  773          1757  1 !
;  774          1758  1 ! IMPLICIT INPUTS:
;  775          1759  1 !       CURRENT_VCB: VCB of volume
;  776          1760  1 !
;  777          1761  1 ! OUTPUT PARAMETERS:
;  778          1762  1 !       NONE
;  779          1763  1 !
;  780          1764  1 ! IMPLICIT OUTPUTS:
;  781          1765  1 !       NONE
;  782          1766  1 !
;  783          1767  1 ! ROUTINE VALUE:
;  784          1768  1 !       1 if successful
;  785          1769  1 !       0 if allocation failure on cache block
;  786          1770  1 !
;  787          1771  1 ! SIDE EFFECTS:
;  788          1772  1 !       quota file FCB hooked into FCB list and quota pointer
;  789          1773  1 !
;  790          1774  1 !--
;  791          1775  1
;  792          1776  2 BEGIN
;  793          1777  2
;  794          1778  2 MAP
;  795          1779  2         FCB             : REF BBLOCK;   ! FCB to hook up
;  796          1780  2
;  797          1781  2 LOCAL
;  798          1782  2         QUOTA_CACHE     : REF BBLOCK,   ! quota cache block allocated
;  799          1783  2         ACB             : REF BBLOCK;   ! AST control block within quota block
;  800          1784  2
;  801          1785  2 BIND_COMMON;
;  802          1786  2
;  803          1787  2 EXTERNAL
;  804          1788  2         SCH$GL_SWPPID   : ADDRESSING_MODE (GENERAL);
;  805          1789  2                                         ! PID of swapper process
;  806          1790  2
;  807          1791  2 EXTERNAL ROUTINE
;  808          1792  2         ALLOCATE        : L_NORM ADDRESSING_MODE (GENERAL), ! allocate system dynamic memory
;  809          1793  2         CACHE_LOCK      : L_NORM,       ! get special cache lock
;  810          1794  2         XQP$UNLOCK_QUOTA : ADDRESSING_MODE (GENERAL);
;  811          1795  2                                         ! release lock with value block
;  812          1796  2
;  813          1797  2
;  814          1798  2 ! Allocate the cache block and link it to the VCB.
;  815          1799  2 !
```

```
;  816          1800  2
;  817          1801  2 QUOTA_CACHE = ALLOCATE (MAXU (.CURRENT_VCB[VCB$W_QUOSIZE], 1) * VCA$C_QUOLENGTH
;  818          1802  2                          + $BYTEOFFSET (VCA$L_QUOLIST), CACHE_TYPE);
;  819          1803  2 IF .QUOTA_CACHE EQL 0
;  820          1804  2 THEN RETURN 0;
;  821          1805  2 QUOTA_CACHE[VCA$W_QUOSIZE] = MAXU (.CURRENT_VCB[VCB$W_QUOSIZE], 1);
;  822          1806  2 CURRENT_VCB[VCB$L_QUOCACHE] = .QUOTA_CACHE;
;  823          1807  2
;  824          1808  2 ! Initialize the AST control blocks in the quota cache header. One is
;  825          1809  2 ! used to post blocking AST's to the swapper to release cache entries.
;  826          1810  2 ! The other is used to trip the cache flush process to flush the entire
;  827          1811  2 ! cache.
;  828          1812  2 !
;  829          1813  2
;  830          1814  2 ACB = QUOTA_CACHE[VCA$B_QUOACB];
;  831          1815  2 ACB[ACB$B_RMOD] = PSL$C_KERNEL + ACB$M_NODELETE;
;  832          1816  2 ACB[ACB$L_PID] = .SCH$GL_SWPPID;
;  833          1817  2 ACB[ACB$L_AST] = XQP$UNLOCK_QUOTA;
;  834          1818  2 ACB = QUOTA_CACHE[VCA$B_QUOFLUSHACB];
;  835          1819  2 ACB[ACB$B_RMOD] = PSL$C_KERNEL + ACB$M_NODELETE;
;  836          1820  2
;  837          1821  2 ! Bump up the access counts in the FCB to show an accessed file.
;  838          1822  2 ! Lock it against truncates.
;  839          1823  2 !
;  840          1824  2
;  841          1825  2 FCB[FCB$W_REFCNT] = .FCB[FCB$W_REFCNT] + 1;
;  842          1826  2 FCB[FCB$W_ACNT] = .FCB[FCB$W_ACNT] + 1;
;  843          1827  2 FCB[FCB$W_TCNT] = .FCB[FCB$W_TCNT] + 1;
;  844          1828  2
;  845          1829  2 ! If the quota file is already write accessed, take out the cache lock
;  846          1830  2 ! on the write access to prevent use of the cache.
;  847          1831  2 !
;  848          1832  2
;  849          1833  2 IF .FCB[FCB$W_WCNT] NEQ 0
;  850          1834  2 AND .BBLOCK [CURRENT_UCB[UCB$L_DEVCHAR2], DEV$V_CLU]
;  851          1835  2 AND .FCB[FCB$L_CACHE[KID] EQL 0
;  852          1836  2 THEN CACHE_LOCK (.FCB[FCB$L_LOCKBASIS], FCB[FCB$L_CACHELKID], 2);
;  853          1837  2
;  854          1838  2 ! Finally enter the quota file pointer in the VCB.
;  855          1839  2 !
;  856          1840  2
;  857          1841  2 CURRENT_VCB[VCB$L_QUOTAFCB] = .FCB;
;  858          1842  2
;  859          1843  2 CLEANUP_FLAGS[CLF_DEACCQFILE] = 1;
;  860          1844  2
;  861          1845  2 RETURN 1;
;  862          1846  2
;  863          1847  1 END;                                    ! end of routine MAKE_QFCB


                                                     .EXTRN  SCH$GL_SWPPID, ALLOCATE
                                                     .EXTRN  CACHE_LOCK, XQP$UNLOCK_QUOTA

                                       0000 00000    .ENTRY  MAKE_QFCB, Save nothing                 ; 1743
                                    06   DD 00002    PUSHL   #6                                      ; 1801
                              50    98   AA  DO 00004    MOVL    -104(BASE), R0                      ;
```

```
                50    60    A0  3C 00008          MOVZWL  96(R0), R0                      ;
                            03  12 0000C          BNEQ    1$                             ;
                50          01  D0 0000E          MOVL    #1, R0                         ;
                50          1C  C4 00011 1$:     MULL2   #28, R0                        ;
                      44    A0  9F 00014          PUSHAB  68(R0)                         ; 1802
00000000G       00          02  FB 00017          CALLS   #2, ALLOCATE                   ;
                51          50  D0 0001E          MOVL    R0, QUOTA_CACHE                ;
                            03  12 00021          BNEQ    2$                             ; 1803
                          0081  31 00023          BRW     5$                             ;
                50    98    AA  D0 00026 2$:     MOVL    -104(BASE), R0                 ; 1805
                50    60    A0  3C 0002A          MOVZWL  96(R0), R0                     ;
                            03  12 0002E          BNEQ    3$                             ;
                50          01  D0 00030          MOVL    #1, R0                         ;
                61          50  B0 00033 3$:     MOVW    R0, (QUOTA_CACHE)              ;
                50    98    AA  D0 00036          MOVL    -104(BASE), R0                 ; 1806
             5C AO          51  D0 0003A          MOVL    QUOTA_CACHE, 92(R0)            ;
                50    0C    A1  9E 0003E          MOVAB   12(R1), ACB                    ; 1814
             0B AO          20  90 00042          MOVB    #32, 11(ACB)                   ; 1815
             0C AO 00000000G 00 D0 00046          MOVL    SCH$GL_SWPPID, 12(ACB)         ; 1816
             10 AO 00000000G 00 9E 0004E          MOVAB   XQP$UNLOCK_QUOTA, 16(ACB)      ; 1817
                50    28    A1  9E 00056          MOVAB   40(R1), ACB                    ; 1818
             0B AO          20  90 0005A          MOVB    #32, 11(ACB)                   ; 1819
                50    04    AC  D0 0005E          MOVL    FCB, R0                        ; 1825
                      18    A0  B6 00062          INCW    24(R0)                         ;
                50    04    AC  D0 00065          MOVL    FCB, R0                        ; 1826
                      1A    A0  B6 00069          INCW    26(R0)                         ;
                50    04    AC  D0 0006C          MOVL    FCB, R0                        ; 1827
                      20    A0  B6 00070          INCW    32(R0)                         ;
                50    04    AC  D0 00073          MOVL    FCB, R0                        ; 1833
                      1C    A0  B5 00077          TSTW    28(R0)                         ;
                            1A  13 0007A          BEQL    4$                             ;
                51    94    AA  D0 0007C          MOVL    -108(BASE), R1                 ; 1834
                12    3C    A1  E9 00080          BLBC    60(R1), 4$                     ;
                      54    A0  D5 00084          TSTL    84(R0)                         ; 1835
                            0D  12 00087          BNEQ    4$                             ;
                            02  DD 00089          PUSHL   #2                             ; 1836
                      54    A0  9F 0008B          PUSHAB  84(R0)                         ;
                      4C    A0  DD 0008E          PUSHL   76(R0)                         ;
             0000G    CF          03  FB 00091          CALLS   #3, CACHE_LOCK         ;
                50    98    AA  D0 00096 4$:     MOVL    -104(BASE), R0                 ; 1841
             54 AO    04    AC  D0 0009A          MOVL    FCB, 84(R0)                    ;
             03 AA          02  88 0009F          BISB2   #2, 3(BASE)                    ; 1843
                50          01  D0 000A3          MOVL    #1, R0                         ; 1845
                            04  000A6             RET                                    ;
                          50  D4 000A7 5$:        CLRL    R0                             ; 1847
                              04 000A9           RET                                    ;

; Routine Size:  170 bytes,    Routine Base:  $CODE$ + 0447

;  864          1848  1
;  865          1849  1 END
;  866          1850  0 ELUDOM
```

```
;                                PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $CODE$                         1265  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       103         0        1000        00:01.9
;
;
;
;                                   COMMAND QUALIFIERS
;
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:QUOTAUTIL/OBJ=OBJ$:QUOTAUTIL MSRC$:QUOTAUTIL/UPDATE=(ENH$:QUOTAUTIL)
;
; Size:          1265 code + 0 data bytes
; Run Time:        01:03.6
; Elapsed Time:    01:58.4
; Lines/CPU Min:    1746
; Lexemes/CPU-Min: 57359
; Memory Used:  321 pages
; Compilation Complete
```

PMS
LIS

QUOTAUTIL
LIS

MAKPTR
LIS

MATCHNAME
LIS

MPWIND
LIS

LOCKDB
LIS

LOCKERS
LIS

MAKACC
LIS

PARSNM
LIS

MAPVBN
LIS

IODONE
LIS

LOCKDN
LIS

MODIFY
LIS

MOUNT
LIS

NXTHDR
LIS

MAKNMB
LIS

MAKSTR
LIS

RWVB
LIS

RDBLOK
LIS

REQUEU
LIS

RWATTR
LIS

REMOVE
LIS

RDHEDR
LIS

RETDIR
LIS