```
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK       KKK    UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK       KKK    UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK       KKK    UUU          UUU  PPP         PPP
BBBBBBBBBBBB     AAA         AAA  CCC            KKKKKKKKK        UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB     AAA         AAA  CCC            KKKKKKKKK        UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB     AAA         AAA  CCC            KKKKKKKKK        UUU          UUU  PPPPPPPPPPPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC            KKK       KKK    UUU          UUU  PPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC            KKK       KKK    UUU          UUU  PPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC            KKK       KKK    UUU          UUU  PPP
BBB         BBB  AAA         AAA  CCC            KKK          KKK  UUU          UUU  PPP
BBB         BBB  AAA         AAA  CCC            KKK          KKK  UUU          UUU  PPP
BBB         BBB  AAA         AAA  CCC            KKK          KKK  UUU          UUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC  KKK          KKK  UUUUUUUUUUUUUUUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC  KKK          KKK  UUUUUUUUUUUUUUUU  PPP
```

```
  SSSSSSSS    AAAAAA    VV      VV  EEEEEEEEEE
  SSSSSSSS    AAAAAA    VV      VV  EEEEEEEEEE
SS           AA      AA  VV      VV  EE
SS           AA      AA  VV      VV  EE
SS           AA      AA  VV      VV  EE
SS           AA      AA  VV      VV  EE
  SSSSSS     AA      AA  VV      VV  EEEEEEEE
  SSSSSS     AA      AA  VV      VV  EEEEEEEE
        SS   AAAAAAAAAA  VV      VV  EE
        SS   AAAAAAAAAA  VV      VV  EE
        SS   AA      AA    VV  VV    EE                ....
        SS   AA      AA    VV  VV    EE                ....
SSSSSSSS     AA      AA      VV      EEEEEEEEEE        ....
SSSSSSSS     AA      AA      VV      EEEEEEEEEE        ....


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

```
;     1        0001  0 MODULE SAVE     (%TITLE 'Save Files-11 media'
;     2        0002  0                  IDENT = 'V04-000'
;     3        0003  0                  ) =
;     4        0004  1 BEGIN
;     5        0005  1
;     6        0006  1 !
;     7        0007  1 !*****************************************************************************
;     8        0008  1 !*                                                                           *
;     9        0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    10        0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    11        0011  1 !*  ALL RIGHTS RESERVED.                                                     *
;    12        0012  1 !*                                                                           *
;    13        0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    14        0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    15        0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    16        0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    17        0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    18        0018  1 !*  TRANSFERRED.                                                             *
;    19        0019  1 !*                                                                           *
;    20        0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    21        0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    22        0022  1 !*  CORPORATION.                                                             *
;    23        0023  1 !*                                                                           *
;    24        0024  1 !*  DIGITAL ASSUMES NO F  PONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    25        0025  1 !*  SOFTWARE ON EQUIPMEN  WHICH IS NOT SUPPLIED BY DIGITAL.                  *
;    26        0026  1 !*                                                                           *
;    27        0027  1 !*                                                                           *
;    28        0028  1 !*****************************************************************************
;    29        0029  1 !
;    30        0030  1
;    31        0031  1 !++
;    32        0032  1 ! FACILITY:
;    33        0033  1 !       Backup/Restore
;    34        0034  1 !
;    35        0035  1 ! ABSTRACT:
;    36        0036  1 !       This module contains the routines that save Files-11 media on
;    37        0037  1 !       backup media.
;    38        0038  1 !
;    39        0039  1 ! ENVIRONMENT:
;    40        0040  1 !       VAX/VMS user mode.
;    41        0041  1 !--
;    42        0042  1 !
;    43        0043  1 ! AUTHOR: M. Jack, CREATION DATE: 02-Sep-1980
;    44        0044  1 !
;    45        0045  1 ! MODIFIED BY:
;    46        0046  1 !
;    47        0047  1 !       V03-023 LY0510          Larry Yetto             19-JUL-1984 08:51
;    48        0048  1 !               Move macros to the top.  Add support for new longword DEVTYP
;    49        0049  1 !               in physical volume attribute record
;    50        0050  1 !
;    51        0051  1 !       V03-022 ACG0435         Andrew C. Goldstein,    11-Jul-1984  17:38
;    52        0052  1 !               Remove references to BRH$L_RESERVED
;    53        0053  1 !
;    54        0054  1 !       V03-021 LMP0272         L. Mark Pilant,         2-Jul-1984  14:31
;    55        0055  1 !               Add support for FIB$L_ACLCTX as well as very long ACLs.
;    56        0056  1 !
;    57        0057  1 !       V03-020 LY0480          Larry Yetto             27-APR-1984 08:32
```

```
;  58          0058  1 !               Surpress the NOBACKUP warning message for system
;  59          0059  1 !               files which are always nobackup
;  60          0060  1 !
;  61          0061  1 !       V03-019 LY0460          Larry Yetto             1-FEB-1984 10:25
;  62          0062  1 !               Add code to make restore operation restartable
;  63          0063  1 !               Add new info messages to notify an operator when a file
;  64          0064  1 !               is marked NOBACKUP and therefore only the file header is
;  65          0065  1 !               saved.
;  66          0066  1 !               Fix compilation warning message about uninitialized variable
;  67          0067  1 !
;  68          0068  1 !       V03-018 ACG0375         Andrew C. Goldstein,    17-Nov-1983  18:32
;  69          0069  1 !               Fix computation of backup summary length, broken by JEP0003
;  70          0070  1 !
;  71          0071  1 !       V03-017 ACG0353         Andrew C. Goldstein,    23-Aug-1983  14:28
;  72          0072  1 !               Fix attribute generation macros to use BEGIN - END blocks,
;  73          0073  1 !               to make their invocation less error prone.
;  74          0074  1 !
;  75          0075  1 !       V03-016 ACG0343         Andrew C. Goldstein,    19-Jul-1983  10:19
;  76          0076  1 !               Use NORECORD option to set backup dates
;  77          0077  1 !
;  78          0078  1 !       V03-015 ACG0340         Andrew C. Goldstein,    22-Jun-1983  18:22
;  79          0079  1 !               Copy last track of MSCP disks in /PHYSICAL operations
;  80          0080  1 !
;  81          0081  1 !       V03-014 JEP0003         J. Eric Pollack,        23-Apr-1983  10:53
;  82          0082  1 !               Add support for encrypted savesets.
;  83          0083  1 !
;  84          0084  1 !       V03-013 ACG0332         Andrew C. Goldstein,    19-Apr-1983  17:59
;  85          0085  1 !               Add support for file highwater mark and RMS journal flags
;  86          0086  1 !
;  87          0087  1 !       V03-012 ACG0327         Andrew C. Goldstein,    11-Apr-1983  15:22
;  88          0088  1 !               Eliminate duplicate summary record when reel switch
;  89          0089  1 !               occurs at start of operation.
;  90          0090  1 !
;  91          0091  1 !       V03-011 ACG0313         Andrew C. Goldstein,    12-Feb-1983  16:56
;  92          0092  1 !               Add routine subtitles
;  93          0093  1 !
;  94          0094  1 !       V03-010 MLJ0104         Martin L. Jack, 26-Jan-1983  16:05
;  95          0095  1 !               Change handling of boot block to allow copying RSX system
;  96          0096  1 !               disks.  Terminate record or delete pass with a message on a
;  97          0097  1 !               writelocked input volume, rather than getting an error for
;  98          0098  1 !               every file.  Avoid record or delete if errors occurred during
;  99          0099  1 !               file processing. Make version string global.
; 100          0100  1 !
; 101          0101  1 !       V03-009 MLJ0100         Martin L. Jack,         06-Jan-1983
; 102          0102  1 !               Fix incorrectly coded tests on length of ident area.
; 103          0103  1 !
; 104          0104  1 !       V03-008 LMP0064         L. Mark Pilant,         15-Dec-1982  9:31
; 105          0105  1 !               Track the changes made to the ACP ACL error handling.  Also,
; 106          0106  1 !               ingnore any ACL processing for ODS-1 files.
; 107          0107  1 !
; 108          0108  1 !       V03-007 MLJ50598        Martin L. Jack, 13-Dec-1982  19:54
; 109          0109  1 !               Repair selected volume copy with verify.
; 110          0110  1 !
; 111          0111  1 !       V03-006 LMP0044         L. Mark Pilant,         21-Oct-1982  14:40
; 112          0112  1 !               Add support for saving and restoring ACL's.
; 113          0113  1 !
; 114          0114  1 !       V03-005 MLJ0096         Martin L. Jack, 24-Aug-1982  15:36
```

```
;  115          0115  1 !               Correct /IMAGE file number conflict bug.  Change to ignore the
;  116          0116  1 !               NOBACKUP attribute for directory files.
;  117          0117  1 !
;  118          0118  1 !       V03-004 MLJ0090         Martin L. Jack, 6-May-1982  8:20
;  119          0119  1 !               Discontinue the requirement that the backup summary record on a
;  120          0120  1 !               continuation volume be in its own block.
;  121          0121  1 !
;  122          0122  1 !       V03-003 MLJ45149        Martin L. Jack, 4-Apr-1982  3:02
;  123          0123  1 !               Correct buffer loss in event of input error.
;  124          0124  1 !
;  125          0125  1 !       V03-002 MLJ0083         Martin L. Jack, 22-Mar-1982  13:25
;  126          0126  1 !               Inhibit saving of scanned but not selected directories under
;  127          0127  1 !               /INTERCHANGE, to save space on distribution media.
;  128          0128  1 !
;  129          0129  1 !       V03-001 MLJ0082         Martin L. Jack, 16-Mar-1982  13:26
;  130          0130  1 !               Avoid writing journal records for directory files to save
;  131          0131  1 !               space.  Avoid file selection tests on directories in a save
;  132          0132  1 !               operation to ensure that needed directories are always saved.
;  133          0133  1 !               Add expanded file specification to NOFILES message.  Correct
;  134          0134  1 !               NOFILES testing so that extra directory copying does not
;  135          0135  1 !               inhibit the message.
;  136          0136  1 !
;  137          0137  1 !       V02-013 MLJ0081         Martin L. Jack, 26-Feb-1982  15:37
;  138          0138  1 !               Implement RETAINMIN and RETAINMAX for new home block fields.
;  139          0139  1 !
;  140          0140  1 !       V02-012 MLJ0075         Martin L. Jack, 28-Jan-1982  20:47
;  141          0141  1 !               Use FIB$V_NORECORD.  Add file version limit handling.
;  142          0142  1 !
;  143          0143  1 !       V02-011 MLJ0063         Martin L. Jack, 22-Dec-1981  3:25
;  144          0144  1 !               Correct loop in GET_PLC_INFO.  Modify operation of /RECORD so
;  145          0145  1 !               that backup dates of saved directories are set only if the
;  146          0146  1 !               directory was selected.  Modify operation of /DELETE so that
;  147          0147  1 !               directories are not deleted.
;  148          0148  1 !
;  149          0149  1 !       V02-010 MLJ0062         Martin L. Jack, 3-Dec-1981  12:24
;  150          0150  1 !               Add DIR_STATUS attribute to support /INCREMENTAL.
;  151          0151  1 !
;  152          0152  1 !       V02-009 MLJ0054         Martin L. Jack, 15-Oct-1981  21:35
;  153          0153  1 !               Implement /VOLUME qualifier.  Combine routines
;  154          0154  1 !               ODS1_VOLUME_ATTRIBUTES and ODS2_VOLUME_ATTRIBUTES, and
;  155          0155  1 !               ODS1_FILE_ATTRIBUTES and ODS2_FILE_ATTRIBUTES.  Compute the
;  156          0156  1 !               implicit NOBACKUP mask for every file so that it is effective
;  157          0157  1 !               without /FAST.  Rework placement data generation, move data
;  158          0158  1 !               structure definitions to COMMON.  Implement /DELETE qualifier.
;  159          0159  1 !               Implement network save sets.  Integrate GET_VM and FREE_VM
;  160          0160  1 !               jacket routines.
;  161          0161  1 !
;  162          0162  1 !       V02-008 MLJ0043         Martin L. Jack, 8-Sep-1981  16:58
;  163          0163  1 !               Account for RMS logical device name change.  Install $GETSYI.
;  164          0164  1 !
;  165          0165  1 !       V02-007 MLJ0036         Martin L. Jack, 29-Aug-1981  16:13
;  166          0166  1 !               Extensive rewriting to complete implementation.
;  167          0167  1 !
;  168          0168  1 !       V02-006 MLJ0025         Martin L. Jack, 8-May-1981  14:09
;  169          0169  1 !               Reorganize qualifier database.  Implement latest version
;  170          0170  1 !               selection.  Add action routine to $PUTMSG for standalone.
;  171          0171  1 !               Make /RECORD restartable.
```

```
;  172          0172  1 !
;  173          0173  1 !       V02-005 MLJ0024         Martin L. Jack, 27-Apr-1981  18:02
;  174          0174  1 !               Correct problem with placement attribute length calculation.
;  175          0175  1 !               Remove dynamic LOCAL_SAVE area, since it can now be local.
;  176          0176  1 !
;  177          0177  1 !       V02-004 MLJ0023         Martin L. Jack, 23-Apr-1981  11:40
;  178          0178  1 !               Implement placement attribute.
;  179          0179  1 !
;  180          0180  1 !       V02-003 MLJ0020         Martin L. Jack, 20-Apr-1981  22:00
;  181          0181  1 !               Implement /JOURNAL qualifier.
;  182          0182  1 !
;  183          0183  1 !       V02-002 MLJ0016         Martin L. Jack, 6-Apr-1981  23:31
;  184          0184  1 !               Always use expanded string as related filename since resultant
;  185          0185  1 !               string may contain an incorrect directory.  Correct error
;  186          0186  1 !               message in SAVE_READ.
;  187          0187  1 !
;  188          0188  1 !       V02-001 MLJ0010         Martin L. Jack, 25-Mar-1981  17:18
;  189          0189  1 !               Reorganize global storage.  Changes for standalone operation.
;  190          0190  1 !               Change "slow" file scan to use BACKUP directory scan logic.
;  191          0191  1 !               Implement image restore.  Implement listing concurrent with
;  192          0192  1 !               another operation.  Ensure that QIO service failures are fatal.
;  193          0193  1 !
;  194          0194  1 !**
```

```
;  196          0195  1 LIBRARY 'SYS$LIBRARY:LIB';
;  197          0196  1 REQUIRE 'LIB$:BACKDEF';
;  198          0646  1 REQUIRE 'SRC$:COMMON';
;  199          1752  1
;  200          1753  1
;  201          1754  1 FORWARD ROUTINE
;  202          1755  1         SAVE_GETBUF:     NOVALUE,       ! Get buffer and initialize header
;  203          1756  1         MAKE_SPACE,                     ! Allocate space in buffer
;  204          1757  1         SAVE_WRITE:      NOVALUE,       ! Release buffer to output routine
;  205          1758  1         SAVE_WAIT:       NOVALUE,       ! Wait for read
;  206          1759  1         SAVE_READ:       NOVALUE,       ! Start asynchronous read
;  207          1760  1         SAVE_READAHEAD:  NOVALUE,       ! Start read-ahead on half of buffers
;  208          1761  1         SAVE_BLOCKS_HANDLER,            ! Restart after reel change
;  209          1762  1         SAVE_BLOCKS,                    ! Driver to save file or volume
;  210          1763  1         BACKUP_SUMMARY:  NOVALUE,       ! Generate backup summary record
;  211          1764  1         FROM_ODS1_DATE:  NOVALUE,       ! Convert ODS-1 date to 64-bit format
;  212          1765  1         ALLOC_PLC_BLOCK:NOVALUE,        ! Allocate placement data block
;  213          1766  1         ALLOC_VBN_BLOCK:NOVALUE,        ! Allocate VBN data block
;  214          1767  1         FREE_PLC_BLOCKS:NOVALUE,        ! Free placement data blocks
;  215          1768  1         GET_PLC_DATA:    NOVALUE,       ! Extract placement data
;  216          1769  1         DUMP_PLC_DATA:   NOVALUE,       ! Dump placement data to record
;  217          1770  1         FILE_ATTRIBUTES:NOVALUE,        ! Generate file attribute record
;  218          1771  1         VOLUME_ATTRIBUTES:
;  219          1772  1                          NOVALUE,       ! Generate volume attribute record
;  220          1773  1         GEN_FID_RECORD:  NOVALUE,       ! Generate file ID record
;  221          1774  1         PHYSVOL_SUMMARY:NOVALUE,        ! Generate physical attribute record
;  222          1775  1         SAVE_VERIFY_REEL:
;  223          1776  1                          NOVALUE,       ! Driver to verify save set
;  224          1777  1         INIT_ATTR:       NOVALUE,       ! Convert header to attributes area
;  225          1778  1         SELECT_INPUT_FILE,              ! Evaluate selection for input file
;  226          1779  1         POSTPROCESS_FILES:
;  227          1780  1                          NOVALUE,       ! Do backup date recording or deletion
;  228          1781  1         SAVE_FILE_ID:    NOVALUE,       ! Save successfully processed file ID
;  229          1782  1         SAVE_ONE_FILE_HANDLER,          ! Recover from BACKUP$_CONTINUE
;  230          1783  1         SAVE_ONE_FILE,                  ! Driver to save one file
;  231          1784  1         PHYS_SAVE:       NOVALUE,       ! Driver for physical save
;  232          1785  1         RESTARTABLE_CONDITION,          ! Test if condition is restartable
;  233          1786  1         SAVE_HANDLER,                   ! Handler for SAVE
;  234          1787  1         SAVE:            NOVALUE;       ! Driver for save
;  235          1788  1
;  236          1789  1
;  237          1790  1 EXTERNAL ROUTINE
;  238          1791  1         LIB$GET_COMMAND:ADDRESSING_MODE(GENERAL),
;  239          1792  1                                         ! Get line from SYS$COMMAND
;  240          1793  1         GET_VM,                         ! Allocate virtual memory
;  241          1794  1         FREE_VM:         NOVALUE,       ! Free virtual memory
;  242          1795  1         ASSIGN_INPUT_CHANNEL,           ! Assign channel to input device
;  243          1796  1         CLOSE_RESTORE:   NOVALUE,       ! Close output file
;  244          1797  1         DEBLOCK:         NOVALUE,       ! Deblock a save set buffer
;  245          1798  1         CRYPTO_INIENC:   NOVALUE WEAK,  ! Init for encryption of saveset
;  246          1799  1         CRYPTO_ENCR_BLOCK:
;  247          1800  1                          NOVALUE WEAK,  ! Encrypt block
;  248          1801  1         EXTRACT_DIR_FILENAME:
;  249          1802  1                          NOVALUE,       ! Extract directory and name
;  250          1803  1         FAST_FILE_SCAN: NOVALUE,        ! Fast file scanner
;  251          1804  1         SLOW_FILE_SCAN: NOVALUE,        ! Slow file scanner
;  252          1805  1         FILE_ERROR:      NOVALUE,       ! Signal file-related error
```

```
;  253          1806  1          FIND_BADBLOCK,                      ! Find an LBN in bad block table
;  254          1807  1          GET_BADBLOCKS,                      ! Get bad block data
;  255          1808  1          INIT_BUFFERS:    NOVALUE,           ! Initialize buffer pool
;  256          1809  1          INIT_RESTORE:    NOVALUE,           ! Initialize a restore or compare
;  257          1810  1          INIT_OUT_SAVE:   NOVALUE,           ! Initialize
;  258          1811  1          REEL_CHECKPOINT:NOVALUE,            ! Checkpoint for reel restart
;  259          1812  1          FIN_OUT_SAVE:    NOVALUE,           ! Terminate
;  260          1813  1          GET_BUFFER,                         ! Get a buffer
;  261          1814  1          FREE_BUFFER:     NOVALUE,           ! Free a buffer
;  262          1815  1          WAIT:            NOVALUE,           ! Wait for I/O completion
;  263          1816  1          MATCH,                              ! Match file specifications
;  264          1817  1          WRITE_BUFFER:    NOVALUE,           ! Write a buffer
;  265          1818  1          CONCURRENT_LIST:NOVALUE,            ! List a buffer
;  266          1819  1          RESTORE:         NOVALUE,           ! Driver for restore module
;  267          1820  1          RESTORE_ONE_RECORD:
;  268          1821  1                           NOVALUE,           ! Restore or compare a buffer
;  269          1822  1          FIN_RESTORE:     NOVALUE,           ! Finish a restore or compare
;  270          1823  1          WRITE_JOUR_SSNAME:
;  271          1824  1                           NOVALUE WEAK,      ! Write journal record for save set
;  272          1825  1          WRITE_JOUR_VOLUME:
;  273          1826  1                           NOVALUE WEAK,      ! Write journal record for volume
;  274          1827  1          WRITE_JOUR_FILE:NOVALUE WEAK,       ! Write journal record for file
;  275          1828  1          WRITE_JOUR_DIRECTORY:
;  276          1829  1                           NOVALUE WEAK,      ! Write final journal record for directory
;  277          1830  1          PUTMSG_ACTRTN:   WEAK,              ! Action routine for standalone $PUTMSG
;  278          1831  1          GET_RESTART:     NOVALUE;           ! Get restart action
;  279          1832  1
;  280          1833  1
;  281          1834  1 EXTERNAL LITERAL
;  282          1835  1          BACKUP$_FACILITY,
;  283          1836  1          BACKUP$_OPENIN,
;  284          1837  1          BACKUP$_OPENOUT,
;  285          1838  1          BACKUP$_CLOSEIN,
;  286          1839  1          BACKUP$_ACTIVEBCB,
;  287          1840  1          BACKUP$_READATTR,
;  288          1841  1          BACKUP$_READERR,
;  289          1842  1          BACKUP$_READBLOCK,
;  290          1843  1          BACKUP$_COPIED,
;  291          1844  1          BACKUP$_CONTINUE,
;  292          1845  1          BACKUP$_GETCHN,
;  293          1846  1          BACKUP$_NOFILES,
;  294          1847  1          BACKUP$_STARTRECORD,
;  295          1848  1          BACKUP$_RECORDSWL,
;  296          1849  1          BACKUP$_WRITEBACK,
;  297          1850  1          BACKUP$_STARTDELETE,
;  298          1851  1          BACKUP$_DELETESWL,
;  299          1852  1          BACKUP$_DELETE,
;  300          1853  1          BACKUP$_FATALERR,
;  301          1854  1          BACKUP$_WRITERRS,
;  302          1855  1          BACKUP$_ACCONFLICT,
;  303          1856  1          BACKUP$_NOBACKUP ,
;  304          1857  1          BACKUP$_HEADCOPIED ;
;  305          1858  1
;  306          1859  1
;  307          1860  1 G$DEFINE();                ! Define global common area
;  308          1861  1
;  309          1862  1
```

```
;  310          1863  1 BUILTIN
;  311          1864  1          TESTBITSC,
;  312          1865  1          INSQUE,
;  313          1866  1          REMQUE,
;  314          1867  1          ROT;
;  315          1868  1
;  316          1869  1 ! Define and create the BACKUP version number string.
;  317          1870  1 !
;  318          1871  1
;  319          1872  1 GLOBAL BIND
;  320          1873  1          VERSION_STRING  = UPLIT BYTE (BACKUP$VERSION, ' (standalone)');
;  321          1874  1
```

```
;  323          1875  1 %SBTTL 'Macros for attribute record handling'
;  324          1876  1 ! Macros to help in building attribute records.
;  325          1877  1 !
;  326          1878  1 MACRO
;  327          1879  1
;  328          1880  1     ! Store "l" bytes from "s".
;  329          1881  1     !
;  330        M 1882  1     STORE_(L,S)=
;  331        M 1883  1         BEGIN
;  332        M 1884  1         %IF L EQL 1 %THEN
;  333        M 1885  1             (.P)<0,8> = S; P = .P + 1 %FI
;  334        M 1886  1         %IF L EQL 2 %THEN
;  335        M 1887  1             (.P)<0,16> = S; P = .P + 2 %FI
;  336        M 1888  1         %IF L EQL 4 %THEN
;  337        M 1889  1             (.P)<0,32> = S; P = .P + 4 %FI
;  338        M 1890  1         %IF L EQL 6 %THEN
;  339        M 1891  1             (.P)<0,32> = .(S)<0,32>; P = .P + 4;
;  340        M 1892  1             (.P)<0,16> = .((S)+4)<0,16>; P = .P + 2 %FI
;  341        M 1893  1         %IF L EQL 8 %THEN
;  342        M 1894  1             (.P)<0,32> = .(S)<0,32>; P = .P + 4;
;  343        M 1895  1             (.P)<0,32> = .((S)+4)<0,32>; P = .P + 4 %FI
;  344        M 1896  1         %IF L EQL 12 %THEN
;  345        M 1897  1             (.P)<0,32> = .(S)<0,32>; P = .P + 4;
;  346        M 1898  1             (.P)<0,32> = .((S)+4)<0,32>; P = .P + 4;
;  347        M 1899  1             (.P)<0,32> = .((S)+8)<0,32>; P = .P + 4 %FI
;  348        M 1900  1         END
;  349          1901  1         %,
;  350          1902  1
;  351          1903  1
;  352          1904  1     ! Store attribute code "n" of length "l" from address "a".
;  353          1905  1     !
;  354        M 1906  1     ATV_(N,L,A)=
;  355        M 1907  1         BEGIN
;  356        M 1908  1         STORE_(2, L);
;  357        M 1909  1         STORE_(2, %NAME('BSA$K_', N));
;  358        M 1910  1         P = CH$MOVE(L, A, .P)
;  359        M 1911  1         END
;  360          1912  1         %,
;  361          1913  1
;  362          1914  1
;  363          1915  1     ! Store device name "n" of length "l" from address "a".
;  364          1916  1     !
;  365        M 1917  1     DVI_(N,L,A)=
;  366        M 1918  1         BEGIN
;  367        M 1919  1         STORE_(2, (L) + 1);
;  368        M 1920  1         STORE_(2, %NAME('BSA$K_', N));
;  369        M 1921  1         P = CH$MOVE(L, A, .P);
;  370        M 1922  1         STORE_(1, %C':')
;  371        M 1923  1         END
;  372          1924  1         %,
;  373          1925  1
;  374          1926  1
;  375          1927  1     ! Store attribute code "n" from "s".
;  376          1928  1     !
;  377        M 1929  1     ATT_(N,S)=
;  378        M 1930  1         BEGIN
;  379        M 1931  1         STORE_(4, %NAME('BSA$S_',N) + %NAME('BSA$K_', N)^16);
```

```
;  380          M 1932  1          STORE_(%NAME('BSA$S_',N), S)
;  381          M 1933  1          END
;  382            1934  1          %;
```

```
;  384          1935  1 %SBTTL 'SAVE_GETBUF - set up new save set buffer'
;  385          1936  1 ROUTINE SAVE_GETBUF: NOVALUE=
;  386          1937  1
;  387          1938  1 !++
;  388          1939  1 !
;  389          1940  1 ! FUNCTIONAL DESCRIPTION:
;  390          1941  1 !       This routine allocates and initializes the file-specific information
;  391          1942  1 !       in a new buffer.
;  392          1943  1 !
;  393          1944  1 ! INPUT PARAMETERS:
;  394          1945  1 !       NONE
;  395          1946  1 !
;  396          1947  1 ! IMPLICIT INPUTS:
;  397          1948  1 !       INPUT_FAB       - Points to the current input FAB.
;  398          1949  1 !       INPUT_NAM       - Points to the current input NAM block.
;  399          1950  1 !       INPUT_NAMEDESC  - Descriptor for current input file name.
;  400          1951  1 !       INPUT_RECATTR   - Record attributes of current input file.
;  401          1952  1 !
;  402          1953  1 ! OUTPUT PARAMETERS:
;  403          1954  1 !       NONE
;  404          1955  1 !
;  405          1956  1 ! IMPLICIT OUTPUTS:
;  406          1957  1 !       INPUT_BCB       - Points to the new buffer.
;  407          1958  1 !
;  408          1959  1 ! ROUTINE VALUE:
;  409          1960  1 !       NONE
;  410          1961  1 !
;  411          1962  1 ! SIDE EFFECTS:
;  412          1963  1 !       NONE
;  413          1964  1 !
;  414          1965  1 !--
;  415          1966  1
;  416          1967  2 BEGIN
;  417          1968  2 LOCAL
;  418          1969  2         BUFFER:         REF BBLOCK;     ! Pointer to buffer
;  419          1970  2
;  420          1971  2
;  421          1972  2 ! Get the new buffer.
;  422          1973  2 !
;  423          1974  2 INPUT_BCB = GET_BUFFER();
;  424          1975  2
;  425          1976  2
;  426          1977  2 ! Check for end of tape.
;  427          1978  2 !
;  428          1979  2 IF .COM_FLAGS[COM_EOV] AND NOT .INPUT_FLAGS[EOV_IN_PROG]
;  429          1980  2 THEN
;  430          1981  3     BEGIN
;  431          1982  3     LOCAL
;  432          1983  3         TEMP:           REF BBLOCK,     ! Temporary pointer to BCB
;  433          1984  3         FIB:            BBLOCK[FIB$C_LENGTH],   ! FIB
;  434          1985  3         FIB_DESC:       VECTOR[2],      ! Descriptor for FIB
;  435          1986  3         STATUS,                         ! Status variable
;  436          1987  3         IOSB:           VECTOR[4,WORD]; ! I/O status block
;  437          1988  3
;  438          1989  3
;  439          1990  3     ! Give back the buffer.  It will be re-acquired later.
;  440          1991  3     !
```

```
;  441          1992  3      INPUT_FLAGS[EOV_IN_PROG] = TRUE;
;  442          1993  3      FREE_BUFFER(.INPUT_BCB);
;  443          1994  3      INPUT_BCB = 0;
;  444          1995  3
;  445          1996  3
;  446          1997  3      ! Clean out the input wait list.  These reads will be reissued later.  The
;  447          1998  3      ! first buffer, however, must be written, because it may contain other
;  448          1999  3      ! information preceding the input buffer.  INPUT_BLOCK will be reset to
;  449          2000  3      ! point to the first unprocessed block.
;  450          2001  3      !
;  451          2002  3      SAVE_WAIT();
;  452          2003  3      UNTIL REMQUE(.INPUT_WAIT[0], TEMP) DO
;  453          2004  4          BEGIN
;  454          2005  4          WAIT(.TEMP);
;  455          2006  4          INPUT_BLOCK = MINU(.INPUT_BLOCK,
;  456          2007  4              .BBLOCK[.TEMP[BCB_RECORD], BRH$L_ADDRESS]);
;  457          2008  4          FREE_BUFFER(.TEMP);
;  458          2009  4          END;
;  459          2010  3
;  460          2011  3
;  461          2012  3      ! Do the physical reel change.
;  462          2013  3      !
;  463          2014  3      IF .QUAL[QUAL_JOUR] THEN WRITE_JOUR_VOLUME();
;  464          2015  3      FIN_OUT_SAVE(TRUE);
;  465          2016  3      COM_FLAGS[COM_EOV] = FALSE;
;  466          2017  3      INPUT_FLAGS[EOV_IN_PROG] = FALSE;
;  467          2018  3      REEL_CHECKPOINT();
;  468          2019  3      INIT_OUT_SAVE(TRUE);
;  469          2020  3      IF .QUAL[QUAL_JOUR] THEN WRITE_JOUR_VOLUME();
;  470          2021  3
;  471          2022  3
;  472          2023  3      ! Write the backup summary record on the new reel. This is done
;  473          2024  3      ! only if save set material has already been written, to prevent
;  474          2025  3      ! a redundant summary record in the boundary case where a reel
;  475          2026  3      ! switch takes place immediately after initializing the save set.
;  476          2027  3      !
;  477          2028  3      IF .RWSV_OUT_SEQ NEQ 0
;  478          2029  3      THEN BACKUP_SUMMARY()
;  479          2030  3      ELSE INPUT_BCB = GET_BUFFER();
;  480          2031  3
;  481          2032  3
;  482          2033  3      ! If there is a reel restart in progress that failed to reposition to
;  483          2034  3      ! the current file, signal a failure.
;  484          2035  3      !
;  485          2036  3      IF TESTBITSC(COM_FLAGS[COM_FAIL_RSTRT])
;  486          2037  3      THEN
;  487          2038  3          FILE_ERROR(
;  488          2039  3              BACKUP$_CONTINUE,           ! See routine SAVE_ONE_FILE_HANDLER
;  489          2040  3              .INPUT_FAB,
;  490          2041  3              .CHKPT_STATUS);
;  491          2042  3
;  492          2043  3
;  493          2044  3      ! If necessary, re-access the file that was accessed at the end of the
;  494          2045  3      ! previous reel.
;  495          2046  3      !
;  496          2047  3      IF .INPUT_FLAGS[INPUT_OPEN] AND .QUAL[QUAL_VER]
;  497          2048  3      THEN
```

```
;  498          2049  4          BEGIN
;  499          2050  4          CH$FILL (0, FIB$C_LENGTH, FIB);
;  500          2051  4          FIB[FIB$L_ACCTL] = FIB$M_NOWRITE OR FIB$M_NORECORD;
;  501          2052  4          IF .INPUT_FLAGS[INPUT_IGNO_INTE] THEN FIB[FIB$L_ACCTL] = FIB$M_NOLOCK OR FIB$M_NORECORD;
;  502          2053  4          FIB[FIB$W_FID_NUM] = .INPUT_NAM[NAM$W_FID_NUM];
;  503          2054  4          FIB[FIB$W_FID_SEQ] = .INPUT_NAM[NAM$W_FID_SEQ];
;  504          2055  4          FIB[FIB$W_FID_RVN] = .INPUT_NAM[NAM$W_FID_RVN];
;  505          2056  4          FIB_DESC[0] = FIB$C_LENGTH;
;  506          2057  4          FIB_DESC[1] = FIB;
;  507        P 2058  4          STATUS = $QIOW(
;  508        P 2059  4              FUNC=IO$_ACCESS OR IO$M_ACCESS,
;  509        P 2060  4              CHAN=.INPUT_CHAN,
;  510        P 2061  4              IOSB=IOSB,
;  511          2062  4              P1=FIB_DESC);
;  512          2063  4          IF .STATUS THEN STATUS = .IOSB[0];
;  513          2064  4          IF NOT .STATUS
;  514          2065  4          THEN
;  515          2066  4              FILE_ERROR(
;  516          2067  4                  BACKUP$_CONTINUE,       ! See routine SAVE_ONE_FILE_HANDLER
;  517          2068  4                  .INPUT_FAB,
;  518          2069  4                  .STATUS);
;  519          2070  3          END;
;  520          2071  3
;  521          2072  3
;  522          2073  3      IF .INPUT_FLAGS[EOV_SAVING]
;  523          2074  3      THEN
;  524          2075  3          SIGNAL(-1);                     ! See routine SAVE_BLOCKS_HANDLER
;  525          2076  2      END;
;  526          2077  2
;  527          2078  2
;  528          2079  2 ! If a file is currently open, put the file-specific information in the
;  529          2080  2 ! buffer header.
;  530          2081  2 !
;  531          2082  2 IF .INPUT_FAB NEQ 0
;  532          2083  2 THEN
;  533          2084  3      BEGIN
;  534          2085  3      BUFFER = .INPUT_BCB[BCB_BUFFER];
;  535          2086  3      BUFFER[BBH$W_FID_NUM] = .INPUT_NAM[NAM$W_FID_NUM];
;  536          2087  3      BUFFER[BBH$W_FID_SEQ] = .INPUT_NAM[NAM$W_FID_SEQ];
;  537          2088  3      BUFFER[BBH$W_FID_RVN] = .INPUT_NAM[NAM$W_FID_RVN];
;  538          2089  3      BUFFER[BBH$W_DID_NUM] = .INPUT_NAM[NAM$W_DID_NUM];
;  539          2090  3      BUFFER[BBH$W_DID_SEQ] = .INPUT_NAM[NAM$W_DID_SEQ];
;  540          2091  3      BUFFER[BBH$W_DID_RVN] = .INPUT_NAM[NAM$W_DID_RVN];
;  541          2092  3      (BUFFER[BBH$T_FI[ENAME])<0,8> = .INPUT_NAMEDESC[DSC$W_LENGTH];
;  542          2093  3      CH$MOVE(
;  543          2094  3          .INPUT_NAMEDESC[DSC$W_LENGTH],
;  544          2095  3          .INPUT_NAMEDESC[DSC$A_POINTER],
;  545          2096  3          BUFFER[BBH$T_FILENAME]+1);
;  546          2097  3      BUFFER[BBH$B_RTYPE] = .INPUT_RECATTR[FAT$B_RTYPE];
;  547          2098  3      BUFFER[BBH$B_RATTRIB] = .INPUT_RECATTR[FAT$B_RATTRIB];
;  548          2099  3      BUFFER[BBH$W_RSIZE] = .INPUT_RECATTR[FAT$W_RSIZE];
;  549          2100  3      BUFFER[BBH$B_BKTSIZE] = .INPUT_RECATTR[FAT$B_BKTSIZE];
;  550          2101  3      BUFFER[BBH$B_VFCSIZE] = .INPUT_RECATTR[FAT$B_VFCSIZE];
;  551          2102  3      BUFFER[BBH$W_MAXREC] = .INPUT_RECATTR[FAT$W_MAXREC];
;  552          2103  3      BUFFER[BBH$L_FILESIZE] = .INPUT_STATBLK[SBK$L_FILESIZE];
;  553          2104  2      END;
;  554          2105  1 END;
```

```
                        .TITLE  SAVE Save Files-11 media
                        .IDENT  \V04-000\

                        .PSECT  COMMON,NOEXE,  OVR,2

00000 GLOBAL_BASE:
                        .BLKB   0
00000 FREE_LIST:
                        .BLKB   8
00008 INPUT_WAIT:
                        .BLKB   8
00010 REREAD_WAIT:
                        .BLKB   8
00018 OUTPUT_WAIT:
                        .BLKB   8
00020 JPI_UIC:.BLKB     4
00024 JPI_USERNAME:
                        .BLKB   12
00030 JPI_DATE:
                        .BLKB   8
00038 JPI_NODE_DESC:
                        .BLKB   8
00040 JPI_CURPRIV:
                        .BLKB   8
00048 SYI_VERSION:
                        .BLKB   4
0004C SYI_SID:.BLKB     4
00050 RWSV_HOLD_LIST:
                        .BLKB   8
00058 RWSV_CRC16:
                        .BLKB   64
00098 RWSV_AUTODIN:
                        .BLKB   64
000D8 RWSV_FILESET_ID:
                        .BLKB   8
000E0 RWSV_VOLUME_ID:
                        .BLKB   12
000EC RWSV_VOL_NUMBER:
                        .BLKB   2
000EE RWSV_SEG_NUMBER:
                        .BLKB   2
000F0 RWSV_FILE_NUMBER:
                        .BLKB   4
000F4 RWSV_SAVE_QUAL:
                        .BLKB   4
000F8 RWSV_SAVE_FAB:
                        .BLKB   4
000FC RWSV_CHAN:
                        .BLKB   4
00100 RWSV_XOR_BCB:
                        .BLKB   4
00104 RWSV_IN_SEQ:
                        .BLKB   4
00108 RWSV_IN_SEQ_0:
                        .BLKB   4
```

```
0010C RWSV_IN_XOR_SEQ:
                .BLKB   4
00110 RWSV_IN_XOR_RFA:
                .BLKB   6
00116 RWSV_LOOKAHEAD:
                .BLKB   1
00117 RWSV_XORSIZE:
                .BLKB   1
00118 RWSV_IN_GROUP_SIZE:
                .BLKB   4
0011C RWSV_IN_ERRORS:
                .BLKB   2
0011E RWSV_IN_XORUSE:
                .BLKB   2
00120 RWSV_IN_ORGERR:
                .BLKB   8
00128 RWSV_IN_VBN:
                .BLKB   4
0012C RWSV_IN_VBN_0:
                .BLKB   4
00130 RWSV_ALLOC:
                .BLKB   4
00134 RWSV_EOF:
                .BLKB   4
00138 RWSV_OUT_SEQ:
                .BLKB   4
0013C RWSV_OUT_VBN:
                .BLKB   4
00140 RWSV_OUT_BLOCK_COUNT:
                .BLKB   4
00144 RWSV_OUT_ERRORS:
                .BLKB   2
00146 RWSV_SEQ_ERRORS:
                .BLKB   2
00148 RWSV_OUT_GROUP_COUNT:
                .BLKB   1
00149 RWSV_PADDING:
                .BLKB   3
0014C QUAL:     .BLKB   112
001BC COM_SSNAME:
                .BLKB   8
001C4 COM_VALID_TYPES:
                .BLKB   2
001C6 COM_FLAGS:
                .BLKB   2
001C8 COM_PADDING:
                .BLKB   1
001C9 COM_BUFF_COUNT:
                .BLKB   1
001CA COM_I_SETCOUNT:
                .BLKB   1
001CB COM_O_SETCOUNT:
                .BLKB   1
001CC COM_I_STRUCNAME:
                .BLKB   12
001D8 COM_O_STRUCNAME:
                .BLKB   12
```

```
001E4 COM_O_BSRDATE:
                .BLKB   8
001EC ALT_SSNAME:
                .BLKB   32
0020C INPUT_FUNC:
                .BLKB   1
0020D INPUT_RTYPE:
                .BLKB   1
0020E OUTPUT_FUNC:
                .BLKB   1
0020F FAST_STRUCLEV:
                .BLKB   1
00210 INPUT_BEG:
                .BLKB   0
00210 INPUT_CHAN:
                .BLKB   4
00214 INPUT_FLAGS:
                .BLKB   2
00216 INPUT_PADDING:
                .BLKB   2
00218 INPUT_FAB:
                .BLKB   4
0021C INPUT_NAM:
                .BLKB   4
00220 INPUT_BCB:
                .BLKB   4
00224 INPUT_QUAL:
                .BLKB   4
00228 INPUT_BAD:
                .BLKB   4
0022C INPUT_BLOCK:
                .BLKB   4
00230 INPUT_MAXBLOCK:
                .BLKB   4
00234 INPUT_MEDIA_ID:
                .BLKB   4
00238 INPUT_NAMEDESC:
                .BLKB   8
00240 INPUT_STATBLK:
                .BLKB   8
00248 INPUT_HDR_BEG:
                .BLKB   0
00248 INPUT_CREDATE:
                .BLKB   8
00250 INPUT_REVDATE:
                .BLKB   8
00258 INPUT_EXPDATE:
                .BLKB   8
00260 INPUT_BAKDATE:
                .BLKB   8
00268 INPUT_FILEOWNER:
                .BLKB   4
0026C INPUT_FILECHAR:
                .BLKB   4
00270 INPUT_RECATTR:
                .BLKB   32
00290 INPUT_HDR_END:
```

```
                        .BLKB   0
00290 INPUT_END:
                        .BLKB   0
00290 INPUT_PROC_LIST:
                        .BLKB   4
00294 INPUT_PLACEMENT:
                        .BLKB   8
0029C INPUT_VBN_LIST:
                        .BLKB   8
002A4 INPUT_PLACE_LEN:
                        .BLKB   2
002A6 INPUT_PADDING_2:
                        .BLKB   2
002A8 OUTPUT_BEG:
                        .BLKB   0
002A8 OUTPUT_CHAN:
                        .BLKB   4
002AC OUTPUT_FLAGS:
                        .BLKB   2
002AE OUTPUT_PADDING:
                        .BLKB   2
002B0 OUTPUT_FAB:
                        .BLKB   4
002B4 OUTPUT_NAM:
                        .BLKB   4
002B8 OUTPUT_BCB:
                        .BLKB   4
002BC OUTPUT_QUAL:
                        .BLKB   4
002C0 OUTPUT_BAD:
                        .BLKB   4
002C4 OUTPUT_BLOCK:
                        .BLKB   4
002C8 OUTPUT_MAXBLOCK:
                        .BLKB   4
002CC OUTPUT_DEVGEOM:
                        .BLKB   8
002D4 OUTPUT_ATTBUF:
                        .BLKB   144
00364 OUTPUT_END:
                        .BLKB   0
00364 LIST_TOTFILES:
                        .BLKB   4
00368 LIST_TOTSIZE:
                        .BLKB   4
0036C VERIFY_FAB:
                        .BLKB   4
00370 VERIFY_USE_COUNT:
                        .BLKB   4
00374 VERIFY_QUAL:
                        .BLKB   4
00378 COMPARE_BCB:
                        .BLKB   4
0037C FAST_BUFFER:
                        .BLKB   4
00380 FAST_BUFFER_SIZE:
                        .BLKB   4
```

```
00384 FAST_RVN:
                .BLKB   1
00385 FAST_PADDING:
                .BLKB   1
00386 DIR_VERLIMIT:
                .BLKB   2
00388 FAST_VOL_BEG:
                .BLKB   0
00388 FAST_IMAP_SIZE:
                .BLKB   4
0038C FAST_IMAP:
                .BLKB   4
00390 FAST_HDR_OFFSET:
                .BLKB   4
00394 FAST_BOOT_LBN:
                .BLKB   4
00398 FAST_VOL_END:
                .BLKB   0
00398 JOUR_BUFFER:
                .BLKB   4
0039C JOUR_DIR:
                .BLKB   4
003A0 JOUR_HIBLK:
                .BLKB   4
003A4 JOUR_EFBLK:
                .BLKB   4
003A8 JOUR_INBLK:
                .BLKB   4
003AC JOUR_FFBYTE:
                .BLKB   2
003AE JOUR_INBYTE:
                .BLKB   2
003B0 JOUR_STRUCT_LEV:
                .BLKB   2
003B2 JOUR_COUNT:
                .BLKB   1
003B3 JOUR_REVERSE:
                .BLKB   1
003B4 JOUR_EXSZ:
                .BLKB   2
003B6 JOUR_PADDING:
                .BLKB   2
003B8 CHKPT_HIGH_SP:
                .BLKB   4
003BC CHKPT_LOW_SP:
                .BLKB   4
003C0 CHKPT_STACK:
                .BLKB   4
003C4 CHKPT_VARS:
                .BLKB   4
003C8 CHKPT_STATUS:
                .BLKB   4
003CC DIR_BEG:.BLKB   0
003CC DIR_CHAN:
                .BLKB   4
003D0 DIR_NAM:.BLKB   4
003D4 DIR_DEV_DESC:
```

```
                        .BLKB   4
003D8 DIR_SEL_DIR:
                        .BLKB   8
003E0 DIR_SEL_NTV:
                        .BLKB   8
003E8 DIR_STRUCLEV:
                        .BLKB   1
003E9 DIR_LEVELS:
                        .BLKB   1
003EA DIR_FLAGS:
                        .BLKB   1
003EB DIR_STATUS:
                        .BLKB   1
003EC DIR_STRING:
                        .BLKB   320
0052C DIR_STACK:
                        .BLKB   612
00790 DIR_SP: .BLKB     4
00794 DIR_SEL_LATEST:
                        .BLKB   4
00798 DIR_END:.BLKB     0
00798 DIR_SCANLIMIT:
                        .BLKB   36
007BC INPUT_MTL:
                        .BLKB   4
007C0 OUTPUT_MTL:
                        .BLKB   4
007C4 CURRENT_MTL:
                        .BLKB   4
007C8 CURRENT_VCB:
                        .BLKB   4
007CC CURRENT_WCB:
                        .BLKB   4
007D0 ACL_FIB_DESCR:
                        .BLKB   8
007D8 ACL_FIB:.BLKB     64
00818 ACL_LENGTH:
                        .BLKB   4
0081C ACL_BUFFER:
                        .BLKB   4
00820 CRYP_IN_CONTEXT:
                        .BLKB   4
00824 CRYP_OU_CONTEXT:
                        .BLKB   4
00828 CRYP_DA_CONTEXT:
                        .BLKB   4
0082C CRYP_DATA_ENCIV:
                        .BLKB   8
00834 CRYP_DATA_CODE:
                        .BLKB   4
00838 CRYP_DATA_KEY:
                        .BLKB   8
00840 CRYP_DATA_IV:
                        .BLKB   8
00848 CRYP_DATA_CKSM:
                        .BLKB   4
```

```
                                                                .PSECT  CODE,NOWRT,2

                                                  30  2E  34  56  00000 P.AAA:  .ASCII  \V4.0\                                  ;
        29  65  6E  6F  6C  61  64  6E  61  74  73  28  20  00004              .ASCII  \ (standalone)\                             ;

                                                                VERSION_STRING==    P.AAA
                                                                .EXTRN  LIB$GET_COMMAND
                                                                .EXTRN  GET_VM, FREE_VM
                                                                .EXTRN  ASSIGN_INPUT_CHANNEL
                                                                .EXTRN  CLOSE_RESTORE, DEBLOCK
                                                                .EXTRN  EXTRACT_DIR_FILENAME
                                                                .EXTRN  FAST_FILE_SCAN, SLOW_FILE_SCAN
                                                                .EXTRN  FILE_ERROR, FIND_BADBLOCK
                                                                .EXTRN  GET_BADBLOCKS, INIT_BUFFERS
                                                                .EXTRN  INIT_RESTORE, INIT_OUT_SAVE
                                                                .EXTRN  REEL_CHECKPOINT
                                                                .EXTRN  FIN_OUT_SAVE, GET_BUFFER
                                                                .EXTRN  FREE_BUFFER, WAIT
                                                                .EXTRN  MATCH, WRITE_BUFFER
                                                                .EXTRN  CONCURRENT_LIST
                                                                .EXTRN  RESTORE, RESTORE_ONE_RECORD
                                                                .EXTRN  FIN_RESTORE, GET_RESTART
                                                                .EXTRN  BACKUP$_FACILITY
                                                                .EXTRN  BACKUP$_OPENIN, BACKUP$_OPENOUT
                                                                .EXTRN  BACKUP$_CLOSEIN
                                                                .EXTRN  BACKUP$_ACTIVEBCB
                                                                .EXTRN  BACKUP$_READATTR
                                                                .EXTRN  BACKUP$_READERR
                                                                .EXTRN  BACKUP$_READBLOCK
                                                                .EXTRN  BACKUP$_COPIED, BACKUP$_CONTINUE
                                                                .EXTRN  BACKUP$_GETCHN, BACKUP$_NOFILES
                                                                .EXTRN  BACKUP$_STARTRECORD
                                                                .EXTRN  BACKUP$_RECORDSWL
                                                                .EXTRN  BACKUP$_WRITEBACK
                                                                .EXTRN  BACKUP$_STARTDELETE
                                                                .EXTRN  BACKUP$_DELETESWL
                                                                .EXTRN  BACKUP$_DELETE, BACKUP$_FATALERR
                                                                .EXTRN  BACKUP$_WRITERRS
                                                                .EXTRN  BACKUP$_ACCONFLICT
                                                                .EXTRN  BACKUP$_NOBACKUP
                                                                .EXTRN  BACKUP$_HEADCOPIED
                                                                .EXTRN  SYS$QIOW
                                                                .WEAK   CRYPTO_INIENC, CRYPTO_ENCR_BLOCK
                                                                .WEAK   WRITE_JOUR_SSNAME
                                                                .WEAK   WRITE_JOUR_VOLUME
                                                                .WEAK   WRITE_JOUR_FILE
                                                                .WEAK   WRITE_JOUR_DIRECTORY
                                                                .WEAK   PUTMSG_ACTRTN

                                                  0FFC 00000 SAVE_GETBUF:
                                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11                    ; 1936
5B 00000000G   00   9E 00002                                    MOVAB   FILE_ERROR, R11                                         ;
5A 00000000G   8F   D0 00009                                    MOVL    #BACKUP$_CONTINUE, R10                                  ;
59 00000000G   00   9E 00010                                    MOVAB   WRITE_JOUR_VOLUME, R9                                   ;
58 00000000G   00   9E 00017                                    MOVAB   FREE_BUFFER, R8                                         ;
57 00000000'   EF   9E 0001E                                    MOVAB   INPUT_FLAGS, R7                                         ;
5E       B0    AE   9E 00025                                    MOVAB   -80(SP), SP                                             ;
```

```
                    00000000G  00              00  FB 000C9          CALLS   #0, GET_BUFFER                                   ; 1974
                           0C  A7              50  D0 00030          MOVL    R0, INPUT_BCB                                   ;
                               03      B2      A7  E8 00034          BLBS    COM_FLAGS, 2$                                   ; 1979
                                           011A  31 00038 1$:        BRW     14$                                             ;
                        F9                 67   01  E0 0003B 2$:    BBS     #1, INPUT_FLAGS, 1$                             ;
                                           67   02  88 0003F         BISB2   #2, INPUT_FLAGS                                 ; 1992
                                               0C  A7  DD 00042      PUSHL   INPUT_BCB                                       ; 1993
                                           68       01  FB 00045     CALLS   #1, FREE_BUFFER                                 ;
                                               0C  A7  D4 00048      CLRL    INPUT_BCB                                       ; 1994
                                 0000V     CF       00  FB 0004B     CALLS   #0, SAVE_WAIT                                   ; 2002
                                           52   FDF4  D7  0F 00050 3$: REMQUE  @INPUT_WAIT, TEMP                            ; 2003
                                                26  1D 00055         BVS     5$                                              ;
                                                52  DD 00057         PUSHL   TEMP                                            ; 2005
                    00000000G  00               01  FB 00059         CALLS   #1, WAIT                                        ;
                                           50       10  A2  D0 00060 MOVL    16(TEMP), R0                                    ; 2007
                                           51       18  A7  D0 00064 MOVL    INPUT_BLOCK, R1                                 ;
                                 08        A0       51  D1 00068     CMPL    R1, 8(R0)                                       ;
                                                    04  1B 0006C     BLEQU   4$                                              ;
                                           51       08  A0  D0 0006E MOVL    8(R0), R1                                       ;
                                 18        A7       51  D0 00072 4$: MOVL    R1, INPUT_BLOCK                                 ; 2006
                                                    52  DD 00076     PUSHL   TEMP                                            ; 2008
                                           68       01  FB 00078     CALLS   #1, FREE_BUFFER                                 ;
                                                    D3  11 0007B     BRB     3$                                              ; 2003
                        03        FF42      C7       06  E1 0007D 5$: BBC     #6, QUAL+10, 6$                                 ; 2014
                                           69       00  FB 00083     CALLS   #0, WRITE_JOUR_VOLUME                           ;
                                                    01  DD 00086 6$: PUSHL   #1                                              ; 2015
                    00000000G  00               01  FB 00088         CALLS   #1, FIN_OUT_SAVE                                ;
                                 B2        A7       01  8A 0008F     BICB2   #1, COM_FLAGS                                   ; 2016
                                           67       02  8A 00093     BICB2   #2, INPUT_FLAGS                                 ; 2017
                    00000000G  00               00  FB 00096         CALLS   #0, REEL_CHECKPOINT                             ; 2018
                                                01  DD 0009D         PUSHL   #1                                              ; 2019
                    00000000G  00               01  FB 0009F         CALLS   #1, INIT_OUT_SAVE                               ;
                        03        FF42      C7       06  E1 000A6     BBC     #6, QUAL+10, 7$                                 ; 2020
                                           69       00  FB 000AC     CALLS   #0, WRITE_JOUR_VOLUME                           ;
                                                FF24  C7  D5 000AF 7$: TSTL    RWSV_OUT_SEQ                                    ; 2028
                                                    07  13 000B3     BEQL    8$                                              ;
                                 0000V     CF       00  FB 000B5     CALLS   #0, BACKUP_SUMMARY                              ; 2029
                                                    0B  11 000BA     BRB     9$                                              ;
                    00000000G  00               00  FB 000BC 8$:    CALLS   #0, GET_BUFFER                                  ; 2030
                           0C  A7               50  D0 000C3         MOVL    R0, INPUT_BCB                                   ;
                        0C        B2        A7       04  E5 000C7 9$: BBCC    #4, COM_FLAGS, 10$                              ; 2036
                                                01B4  C7  DD 000CC    PUSHL   CHKPT_STATUS                                    ; 2041
                                                   04  A7  DD 000D0    PUSHL   INPUT_FAB                                       ; 2040
                                                    5A  DD 000D3     PUSHL   R10                                             ; 2038
                                           6B       03  FB 000D5     CALLS   #3, FILE_ERROR                                  ;
                                           6C       67  E9 000D8 10$: BLBC    INPUT_FLAGS, 13$                                ; 2047
                                                FF45  C7  95 000DB     TSTB    QUAL+13                                         ;
                                                    66  18 000DF     BGEQ    13$                                             ;
0040    8F                  00              6E       00  2C 000E1     MOVC5   #0, (SP), #0, #64, FIB                          ; 2050
                                                10  AE      000E8                                                             ;
                                     10     AE 00200001  8F  D0 000EA MOVL    #2097153, FIB                                   ; 2051
                        08                  67       04  E1 000F2     BBC     #4, INPUT_FLAGS, 11$                            ; 2052
                                     10     AE 00300000  8F  D0 000F6 MOVL    #3145728, FIB                                   ;
                                           50       08  A7  D0 000FE 11$: MOVL    INPUT_NAM, R0                           ; 2053
                                     14     AE       24  A0  D0 00102 MOVL    36(R0), FIB+4                                   ;
                                     18     AE       28  A0  B0 00107 MOVW    40(R0), FIB+8                                   ; 2055
                                     08     AE       40  8F  9A 0010C MOVZBL  #64, FIB_DESC                                   ; 2056
```

```
                          0C    AE        10    AE   9E 00111          MOVAB   FIB, FIB_DESC+4                          ; 2057
                                          7E    7C 00116          CLRQ    -(SP)                                    ; 2062
                                          7E    7C 00118          CLRQ    -(SP)                                    ;
                                          7E    D4 0011A          CLRL    -(SP)                                    ;
                                    1C    AE    9F 0011C          PUSHAB  FIB_DESC                                 ;
                                          7E    7C 0011F          CLRQ    -(SP)                                    ;
                                    20    AE    9F 00121          PUSHAB  IOSB                                     ;
                                 7E       72    8F 9A 00124       MOVZBL  #114, -(SP)                              ;
                                    FC    A7    DD 00128          PUSHL   INPUT_CHAN                               ;
                                          7E    D4 0012B          CLRL    -(SP)                                    ;
                       00000000G  00              0C FB 0012D       CALLS   #12, SYS$QIOW                            ;
                                 06              50 E9 00134       BLBC    STATUS, 12$                              ; 2063
                                 50              6E 3C 00137       MOVZWL  IOSB, STATUS                             ;
                                 0A              50 E8 0013A       BLBS    STATUS, 13$                              ; 2064
                                                 50 DD 0013D 12$:  PUSHL   STATUS                                   ; 2069
                                    04    A7    DD 0013F          PUSHL   INPUT_FAB                                ; 2068
                                          5A    DD 00142          PUSHL   R10                                      ; 2066
                                 6B              03 FB 00144       CALLS   #3, FILE_ERROR                           ;
                          0A     67              02 E1 00147 13$:  BBC     #2, INPUT_FLAGS, 14$                     ; 2073
                                 7E              01 CE 0014B       MNEGL   #1, -(SP)                                ; 2075
                       00000000G  00              01 FB 0014E       CALLS   #1, LIB$SIGNAL                           ;
                                    04    A7    D5 00155 14$:     TSTL    INPUT_FAB                                ; 2082
                                          34    13 00158          BEQL    15$                                      ;
                                 50       0C    A7 D0 0015A       MOVL    INPUT_BCB, R0                            ; 2085
                                 56       0C    A0 D0 0015E       MOVL    12(R0), BUFFER                           ;
                                 50       08    A7 D0 00162       MOVL    INPUT_NAM, R0                            ; 2086
                              50    A6    24    A0 7D 00166       MOVQ    36(R0), 80(BUFFER)                       ;
                              58    A6    2C    A0 D0 0016B       MOVL    44(R0), 88(BUFFER)                       ; 2090
                              5C    A6    24    A7 90 00170       MOVB    INPUT_NAMEDESC, 92(BUFFER)               ; 2092
                        5D    A6    28    B7    24 A7 28 00175    MOVC3   INPUT_NAMEDESC, @INPUT_NAMEDESC+4, -     ; 2096
                                                                          93(BUFFER)                               ;
                           00DC    C6    5C    A7 D0 0017C       MOVL    INPUT_RECATTR, 220(BUFFER)               ; 2097
                           00E0    C6    6A    A7 D0 00182       MOVL    INPUT_RECATTR+14, 224(BUFFER)            ; 2100
                           00E4    C6    30    A7 D0 00188       MOVL    INPUT_STATBLK+4, 228(BUFFER)             ; 2103
                                                 04 0018E 15$:     RET                                              ; 2105

; Routine Size:  399 bytes,    Routine Base:  CODE + 0011
```

```
;   556          2106  1 %SBTTL 'MAKE_SPACE - allocate space in save set buffer'
;   557          2107  1 ROUTINE MAKE_SPACE(N)=
;   558          2108  1
;   559          2109  1 !++
;   560          2110  1 !
;   561          2111  1 ! FUNCTIONAL DESCRIPTION:
;   562          2112  1 !       This routine allocates space in a save set buffer.
;   563          2113  1 !
;   564          2114  1 ! INPUT PARAMETERS:
;   565          2115  1 !       N               - Number of bytes to allocate.
;   566          2116  1 !
;   567          2117  1 ! IMPLICIT INPUTS:
;   568          2118  1 !       INPUT_BCB       - Buffer control block for current buffer.
;   569          2119  1 !
;   570          2120  1 ! OUTPUT PARAMETERS:
;   571          2121  1 !       NONE
;   572          2122  1 !
;   573          2123  1 ! IMPLICIT OUTPUTS:
;   574          2124  1 !       INPUT_BCB       - Updated as appropriate.
;   575          2125  1 !
;   576          2126  1 ! ROUTINE VALUE:
;   577          2127  1 !       Pointer to the allocated space.
;   578          2128  1 !
;   579          2129  1 ! SIDE EFFECTS:
;   580          2130  1 !       If there is insufficient space in the current buffer, it is released to
;   581          2131  1 !       the output procedure and a new buffer allocated.
;   582          2132  1 !
;   583          2133  1 !--
;   584          2134  1
;   585          2135  2 BEGIN
;   586          2136  2 LOCAL
;   587          2137  2         AREA;                   ! Pointer to allocated space
;   588          2138  2
;   589          2139  2
;   590          2140  2 IF .INPUT_BCB EQL 0
;   591          2141  2 THEN
;   592          2142  2     SAVE_GETBUF()
;   593          2143  2 ELSE
;   594          2144  2     IF
;   595          2145  2         .INPUT_BCB[BCB_RECORD] + .N GTRU
;   596          2146  2         .INPUT_BCB[BCB_BUFFER] + .INPUT_BCB[BCB_SIZE]
;   597          2147  2     THEN
;   598          2148  3         BEGIN
;   599          2149  3         !
;   600          2150  3         ! Insufficient space in current buffer.  Write the old buffer, and get
;   601          2151  3         ! another one.
;   602          2152  3         !
;   603          2153  3         SAVE_WRITE();
;   604          2154  3         SAVE_GETBUF();
;   605          2155  2         END;
;   606          2156  2
;   607          2157  2
;   608          2158  2 ! Update the record pointer, and return the address of the record.
;   609          2159  2 !
;   610          2160  2 AREA = .INPUT_BCB[BCB_RECORD];
;   611          2161  2 INPUT_BCB[BCB_RECORD] = .INPUT_BCB[BCB_RECORD] + .N;
;   612          2162  2 .AREA
```

```
;  613          2163  1 END;


                                        000C 00000 MAKE_SPACE:
                                                           .WORD   Save R2,R3                              ; 2107
                          53 00000000'  EF  9E 00002       MOVAB   INPUT_BCB, R3                          ;
                          52            63  D0 00009       MOVL    INPUT_BCB, R2                          ; 2140
                                        19  13 0000C       BEQL    1$                                     ;
                50    10  A2      04    AC  C1 0000E       ADDL3   N, 16(R2), R0                          ; 2145
                          51      08    A2  3C 00014       MOVZWL  8(R2), R1                              ; 2146
                52        51      0C    A2  C1 00018       ADDL3   12(R2), R1, R2                         ;
                          52            50  D1 0001D       CMPL    R0, R2                                 ;
                                        0A  1B 00020       BLEQU   2$                                     ;
                    0000V CF            00  FB 00022       CALLS   #0, SAVE_WRITE                         ; 2153
                    FE45  CF            00  FB 00027 1$:   CALLS   #0, SAVE_GETBUF                        ; 2154
                          51            63  D0 0002C 2$:   MOVL    INPUT_BCB, R1                          ; 2160
                          50      10    A1  D0 0002F       MOVL    16(R1), AREA                           ;
                      10  A1      04    AC  C0 00033       ADDL2   N, 16(R1)                              ; 2161
                                        04  00038          RET                                            ; 2163

; Routine Size:  57 bytes,    Routine Base:  CODE + 01A0
```

```
;  615          2164  1 %SBTTL 'SAVE_WRITE - write save set buffer'
;  616          2165  1 ROUTINE SAVE_WRITE: NOVALUE=
;  617          2166  1
;  618          2167  1 !++
;  619          2168  1 !
;  620          2169  1 ! FUNCTIONAL DESCRIPTION:
;  621          2170  1 !       This routine passes the current buffer to the appropriate output
;  622          2171  1 !       procedure.
;  623          2172  1 !
;  624          2173  1 ! INPUT PARAMETERS:
;  625          2174  1 !       NONE
;  626          2175  1 !
;  627          2176  1 ! IMPLICIT INPUTS:
;  628          2177  1 !       INPUT_BCB       - Pointer to current buffer.
;  629          2178  1 !
;  630          2179  1 ! OUTPUT PARAMETERS:
;  631          2180  1 !       NONE
;  632          2181  1 !
;  633          2182  1 ! IMPLICIT OUTPUTS:
;  634          2183  1 !       INPUT_BCB       - Zeroed to indicate no current buffer.
;  635          2184  1 !
;  636          2185  1 ! ROUTINE VALUE:
;  637          2186  1 !       NONE
;  638          2187  1 !
;  639          2188  1 ! SIDE EFFECTS:
;  640          2189  1 !       Buffer passed to output procedure.
;  641          2190  1 !
;  642          2191  1 !--
;  643          2192  1
;  644          2193  2 BEGIN
;  645          2194  2 LOCAL
;  646          2195  2         BCB:            REF BBLOCK,     ! Pointer to BCB
;  647          2196  2         BUFFER,                         ! Pointer to buffer
;  648          2197  2         RECP:           REF BBLOCK;     ! Pointer to record
;  649          2198  2
;  650          2199  2
;  651          2200  2 BCB = .INPUT_BCB;
;  652          2201  2 IF .BCB NEQ 0
;  653          2202  2 THEN
;  654          2203  3     BEGIN
;  655          2204  3
;  656          2205  3     ! Put in the trailing null record if required.
;  657          2206  3     !
;  658          2207  3     BUFFER = .BCB[BCB_BUFFER];
;  659          2208  3     RECP = .BCB[BCB_RECORD];
;  660          2209  3     IF .RECP LEQA .BUFFER+.BCB[BCB_SIZE]-BRH$K_LENGTH
;  661          2210  3     THEN
;  662          2211  4         BEGIN
;  663          2212  4         CH$FILL(0, .BUFFER+.BCB[BCB_SIZE]-.RECP, .RECP);
;  664          2213  4         RECP[BRH$W_RSIZE] = .BUFFER+.BCB[BCB_SIZE]-.RECP-BRH$K_LENGTH;
;  665          2214  4         RECP[BRH$W_RTYPE] = BRH$K_NULL;
;  666          2215  3         END;
;  667          2216  3
;  668          2217  3
;  669          2218  3     ! Indicate no current buffer.
;  670          2219  3     !
;  671          2220  3     INPUT_BCB = 0;
```

```
;  672          2221  3
;  673          2222  3
;  674          2223  3      ! Pass the buffer to the output routine.
;  675          2224  3      !
;  676          2225  3      IF .QUAL[QUAL_LIST]
;  677          2226  3          THEN CONCURRENT_LIST(.BCB);
;  678          2227  3      IF .QUAL[QUAL_OSAV]
;  679          2228  3          THEN
;  680          2229  4          BEGIN
;  681          2230  4          IF .QUAL[QUAL_SS_ENCRYP] AND CRYPTO_ENCR_BLOCK NEQA 0
;  682          2231  4                  THEN CRYPTO_ENCR_BLOCK (.BCB)
;  683          2232  4                  ELSE WRITE_BUFFER(.BCB);
;  684          2233  4          END
;  685          2234  3          ELSE DEBLOCK(.BCB, RESTORE_ONE_RECORD);
;  686          2235  2      END;
;  687          2236  1 END;

                                07FC 00000 SAVE_WRITE:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10                ; 2165
                    5A 00000000G   00  9E 00002    MOVAB   CRYPTO_ENCR_BLOCK, R10                  ;
                    59 00000000'   EF  9E 00009    MOVAB   INPUT_BCB, R9                           ;
                    57             69  D0 00010    MOVL    INPUT_BCB, BCB                          ; 2200
                                   69  13 00013    BEQL    5$                                      ; 2201
                    50       0C    A7  D0 00015    MOVL    12(BCB), BUFFER                         ; 2207
                    56       10    A7  D0 00019    MOVL    16(BCB), RECP                           ; 2208
                    51       08    A7  3C 0001D    MOVZWL  8(BCB), R1                              ; 2209
                 58 50             51  C1 00021    ADDL3   R1, BUFFER, R8                          ;
                    50       F0    A8  9E 00025    MOVAB   -16(R8), R0                             ;
                    50             56  D1 00029    CMPL    RECP, R0                                ;
                                   10  1A 0002C    BGTRU   1$                                      ;
                    58             56  C2 0002E    SUBL2   RECP, R8                                ; 2212
                 58 00             6E  00  2C 00031    MOVC5   #0, (SP), #0, R8, (RECP)            ;
                                       66  00036    
                    66             58  10  A3 00037    SUBW3   #16, R8, (RECP)                     ; 2213
                                 02    A6  B4 0003B    CLRW    2(RECP)                             ; 2214
                                       69  D4 0003E 1$: CLRL    INPUT_BCB                           ; 2220
                                 09 FF37 C9  E9 00040    BLBC    QUAL+T1, 2$                         ; 2225
                                       57  DD 00045    PUSHL   BCB                                 ; 2226
                          00000000G    00  01  FB 00047    CALLS   #1, CONCURRENT_LIST             ;
                                    FF3B C9  95 0004E 2$: TSTB    QUAL+15                         ; 2227
                                       1B  18 00052    BGEQ    4$                                  ;
                    0B       FF3A C9      04  E1 00054    BBC     #4, QUAL+14, 3$                 ; 2230
                                   50      6A  9E 0005A    MOVAB   CRYPTO_ENCR_BLOCK, R0           ;
                                       06  13 0005D    BEQL    3$                                  ;
                                       57  DD 0005F    PUSHL   BCB                                 ; 2231
                                   6A      01  FB 00061    CALLS   #1, CRYPTO_ENCR_BLOCK           ;
                                       04  00064    RET                                         ;
                                       57  DD 00065 3$: PUSHL   BCB                                 ; 2232
                          00000000G    00  01  FB 00067    CALLS   #1, WRITE_BUFFER                ;
                                       04  0006E    RET                                         ; 2227
                                00000000G  00  9F 0006F 4$: PUSHAB  RESTORE_ONE_RECORD             ; 2234
                                       57  DD 00075    PUSHL   BCB                                 ;
                          00000000G    00  02  FB 00077    CALLS   #2, DEBLOCK                     ;
```

```
                                        04 0007E 5$:      RET                                                 ; 2236

; Routine Size:  127 bytes,    Routine Base:  CODE + 01D9
```

```
;  689     2237  1 %SBTTL 'SAVE_WAIT - wait for pending read'
;  690     2238  1 ROUTINE SAVE_WAIT: NOVALUE=
;  691     2239  1
;  692     2240  1 !++
;  693     2241  1 !
;  694     2242  1 ! FUNCTIONAL DESCRIPTION:
;  695     2243  1 !       This routine waits for completion of the first pending read and
;  696     2244  1 !       passes it to the next operation.
;  697     2245  1 !
;  698     2246  1 ! INPUT PARAMETERS:
;  699     2247  1 !       NONE
;  700     2248  1 !
;  701     2249  1 ! IMPLICIT INPUTS:
;  702     2250  1 !       NONE
;  703     2251  1 !
;  704     2252  1 ! OUTPUT PARAMETERS:
;  705     2253  1 !       NONE
;  706     2254  1 !
;  707     2255  1 ! IMPLICIT OUTPUTS:
;  708     2256  1 !       NONE
;  709     2257  1 !
;  710     2258  1 ! ROUTINE VALUE:
;  711     2259  1 !       NONE
;  712     2260  1 !
;  713     2261  1 ! SIDE EFFECTS:
;  714     2262  1 !       NONE
;  715     2263  1 !
;  716     2264  1 !--
;  717     2265  1
;  718     2266  2 BEGIN
;  719     2267  2 LOCAL
;  720     2268  2         BLOCKS,                         ! Blocks in current transfer
;  721     2269  2         TEMP,                           ! Temporary BCB pointer
;  722     2270  2         REC:            REF BBLOCK,     ! Record header pointer for transfer
;  723     2271  2         STATUS;                         ! Status return
;  724     2272  2
;  725     2273  2
;  726     2274  2 UNTIL REMQUE(.INPUT_WAIT[0], INPUT_BCB) DO
;  727     2275  3     BEGIN
;  728     2276  3     WAIT(.INPUT_BCB);
;  729     2277  3
;  730     2278  3
;  731     2279  3     ! Point to the record, and get the number of blocks in the transfer.
;  732     2280  3     !
;  733     2281  3     REC = .INPUT_BCB[BCB_RECORD];
;  734     2282  3     BLOCKS = .REC[BRH$W_RSIZE] / 512;
;  735     2283  3
;  736     2284  3
;  737     2285  3     IF .INPUT_BCB[BCB_IO_STATUS]
;  738     2286  3     THEN
;  739     2287  4         BEGIN
;  740     2288  4         !
;  741     2289  4         ! Transfer was successful.  Advance the record pointer in the BCB
;  742     2290  4         ! past the transfer.  Unless this is the last transfer, which is
;  743     2291  4         ! the case if it includes the last block to be transferred, the
;  744     2292  4         ! buffer is full and should be passed to the output procedure.  If
;  745     2293  4         ! it is the last transfer, return with the remaining space in the
```

```
;  746          2294  4          ! buffer available for further use.
;  747          2295  4          !
;  748          2296  4          INPUT_BCB[BCB_RECORD] = .REC + .REC[BRH$W_RSIZE] + BRH$C_LENGTH;
;  749          2297  4          IF .REC[BRH$L_ADDRESS] + .BLOCKS LEQU .INPUT_MAXBLOCK
;  750          2298  4              THEN SAVE_WRITE();
;  751          2299  4          RETURN;
;  752          2300  4          END
;  753          2301  3      ELSE
;  754          2302  4          BEGIN
;  755          2303  4          !
;  756          2304  4          ! Transfer was unsuccessful.  Hang onto this buffer, but clear out
;  757          2305  4          ! the remaining entries in the input wait list.  All of these
;  758          2306  4          ! transfers will have to be issued again.
;  759          2307  4          !
;  760          2308  4          UNTIL REMQUE(.INPUT_WAIT[0], TEMP) DO
;  761          2309  5              BEGIN
;  762          2310  5              WAIT(.TEMP);
;  763          2311  5              FREE_BUFFER(.TEMP);
;  764          2312  4              END;
;  765          2313  4
;  766          2314  4
;  767          2315  4          ! Reset the input block counter to the beginning of the failed
;  768          2316  4          ! transfer.  Redo the transfer one block at a time noting the
;  769          2317  4          ! blocks that fail.
;  770          2318  4          !
;  771          2319  4          INPUT_BLOCK = .REC[BRH$L_ADDRESS];
;  772          2320  4          INCR XBLOCK FROM .INPUT_BLOCK TO .INPUT_BLOCK + .BLOCKS - 1 DO
;  773          2321  5              BEGIN
;  774          2322  5              REC = MAKE_SPACE(512 + BRH$C_LENGTH);
;  775          2323  5              REC[BRH$W_RSIZE] = 512;
;  776          2324  5              REC[BRH$W_RTYPE] = .INPUT_RTYPE;
;  777          2325  5              REC[BRH$L_FLAGS] = 0;
;  778          2326  5              IF .INPUT_FILECHAR[FCH$V_DIRECTORY]
;  779          2327  5                  THEN REC[BRH$V_DIRECTORY] = TRUE;
;  780          2328  5              REC[BRH$L_ADDRESS] = .XBLOCK;
;  781          2329  5              $ASSUME ($BYTEOFFSET (BRH$W_BLOCKFLAGS) + 4, EQL, BRH$K_LENGTH);
;  782          2330  5              (REC[BRH$W_BLOCKFLAGS])<0,32> = 0;
;  783        P 2331  5              STATUS = $QIOW(
;  784        P 2332  5                  FUNC=.INPUT_FUNC,
;  785        P 2333  5                  CHAN=.INPUT_CHAN,
;  786        P 2334  5                  IOSB=INPUT_BCB[BCB_IOSB],
;  787        P 2335  5                  P1=.REC + BRH$C_LENGTH,
;  788        P 2336  5                  P2=512,
;  789          2337  5                  P3=.XBLOCK);
;  790          2338  5              IF .STATUS THEN STATUS = .INPUT_BCB[BCB_IO_STATUS];
;  791          2339  5              IF NOT .STATUS THEN IF NOT FIND_BADBLOCK(.INPUT_BAD, .XBLOCK)
;  792          2340  5              THEN
;  793          2341  6                  BEGIN
;  794          2342  6                  FILE_ERROR(
;  795          2343  6                      BACKUP$_READBLOCK,
;  796          2344  6                      .INPUT_FAB,
;  797          2345  6                      .XBLOCK,
;  798          2346  6                      .STATUS);
;  799          2347  6                  REC[BRH$V_BADDATA] = TRUE;
;  800          2348  5                  END;
;  801          2349  4              END;
;  802          2350  4
```

```
;  803          2351  4
;  804          2352  4          ! The entire failed transfer has now been redone.  Pass the buffer to
;  805          2353  4          ! the output procedure.
;  806          2354  4          !
;  807          2355  4          INPUT_BCB[BCB_RECORD] = .REC + 512 + BRH$C_LENGTH;
;  808          2356  4          SAVE_WRITE();
;  809          2357  4
;  810          2358  4          ! Advance the input block pointer past this transfer, and restart the
;  811          2359  4          ! readahead.
;  812          2360  4          !
;  813          2361  4          INPUT_BLOCK = .INPUT_BLOCK + .BLOCKS;
;  814          2362  4          IF NOT .INPUT_FLAGS[EOV_IN_PROG] THEN SAVE_READAHEAD();
;  815          2363  3          END;
;  816          2364  2      END;
;  817          2365  2  INPUT_BCB = 0;
;  818          2366  2  INPUT_BCB = 0;
;  819          2367  1  END;


                                03FC 00000 SAVE_WAIT:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                    ; 2238
59 00000000G   00   9E 00002               MOVAB   WAIT, R9                                        ;
58 00000000'   EF   9E 00009               MOVAB   INPUT_BCB, R8                                   ;
68       FDE8  D8   0F 00010 1$:            REMQUE  @INPUT_WAIT, INPUT_BCB                          ; 2274
               03   1C 00015               BVC     2$                                              ;
             0107   31 00017               BRW     13$                                             ;
               68   DD 0001A 2$:            PUSHL   INPUT_BCB                                       ; 2276
          69        01   FB 0001C           CALLS   #1, WAIT                                        ;
          50        68   D0 0001F           MOVL    INPUT_BCB, R0                                   ; 2281
          52    10  A0   D0 00022           MOVL    16(R0), REC                                     ;
          51        62   3C 00026           MOVZWL  (REC), R1                                       ; 2282
55        51 00000200  8F   C7 00029        DIVL3   #512, R1, BLOCKS                                ;
          18    18  A0   E9 00031           BLBC    24(R0), 4$                                      ; 2285
      10  A0    10 A142  9E 00035           MOVAB   16(R1)[REC], 16(R0)                             ; 2296
50    55    08  A2   C1 0003B               ADDL3   8(REC), BLOCKS, R0                              ; 2297
      10  A8        50   D1 00040           CMPL    R0, INPUT_MAXBLOCK                              ;
               01   1B 00044               BLEQU   3$                                              ;
               04   00046                   RET                                                     ;
    FF35  CF         00   FB 00047 3$:      CALLS   #0, SAVE_WRITE                                  ; 2298
               04   0004C                   RET                                                     ; 2287
          56   FDE8  D8   0F 0004D 4$:      REMQUE  @INPUT_WAIT, TEMP                               ; 2308
               10   1D 00052               BVS     5$                                              ;
               56   DD 00054               PUSHL   TEMP                                            ; 2310
          69        01   FB 00056           CALLS   #1, WAIT                                        ;
               56   DD 00059               PUSHL   TEMP                                            ; 2311
00000000G      00   01   FB 0005B           CALLS   #1, FREE_BUFFER                                 ;
               E9   11 00062               BRB     4$                                              ; 2308
      0C  A8   08   A2   D0 00064 5$:      MOVL    8(REC), INPUT_BLOCK                             ; 2319
57    55    0C  A8   C1 00069               ADDL3   INPUT_BLOCK, BLOCKS, R7                         ; 2320
53    0C  A8        01   C3 0006E           SUBL3   #1, INPUT_BLOCK, XBLOCK                         ;
             0083   31 00073               BRW     9$                                              ;
      7E    0210  8F   3C 00076 6$:         MOVZWL  #528, -(SP)                                     ; 2322
    FEC8  CF         01   FB 0007B         CALLS   #1, MAKE_SPACE                                  ;
          52         50   D0 00080         MOVL    R0, REC                                         ;
```

```
                        62      0200    8F  B0 00083          MOVW    #512, (REC)                             ; 2323
                    02  A2        ED    A8  9B 00088          MOVZBW  INPUT_RTYPE, 2(REC)                     ; 2324
                        04      A2      D4  0008D             CLRL    4(REC)                                  ; 2325
                04  4D  A8              05  E1 00090          BBC     #5, INPUT_FILECHAR+1, 7$                ; 2326
                    04  A2              02  88 00095          BISB2   #2, 4(REC)                              ; 2327
                    08  A2              53  D0 00099 7$:      MOVL    XBLOCK, 8(REC)                          ; 2328
                        0C      A2      D4  0009D             CLRL    12(REC)                                 ; 2330
                                        7E  7C 000A0          CLRQ    -(SP)                                   ; 2337
                                        7E  D4 000A2          CLRL    -(SP)
                                        53  DD 000A4          PUSHL   XBLOCK
                        7E      0200    8F  3C 000A6          MOVZWL  #512, -(SP)
                                10      A2  9F 000AB          PUSHAB  16(REC)
                                        7E  7C 000AE          CLRQ    -(SP)
                7E              68          18  C1 000B0      ADDL3   #24, INPUT_BCB, -(SP)
                        7E          EC      A8  9A 000B4      MOVZBL  INPUT_FUNC, -(SP)
                                    F0      A8  DD 000B8      PUSHL   INPUT_CHAN
                                            7E  D4 000BB      CLRL    -(SP)
        00000000G               00          0C  FB 000BD      CALLS   #12, SYS$QIOW
                                54          50  D0 000C4      MOVL    R0, STATUS
                                0A          54  E9 000C7      BLBC    STATUS, 8$                              ; 2338
                                50          68  D0 000CA      MOVL    INPUT_BCB, R0
                                54      18  A0  3C 000CD      MOVZWL  24(R0), STATUS
                                25          54  E8 000D1      BLBS    STATUS, 9$                              ; 2339
                                            53  DD 000D4 8$:  PUSHL   XBLOCK
                                    08      A8  DD 000D6      PUSHL   INPUT_BAD
        00000000G               00          02  FB 000D9      CALLS   #2, FIND_BADBLOCK
                                16          50  E8 000E0      BLBS    R0, 9$
                                            18  BB 000E3      PUSHR   #^M<R3,R4>                              ; 2345
                                    F8      A8  DD 000E5      PUSHL   INPUT_FAB                               ; 2344
                            00000000G       8F  DD 000E8      PUSHL   #BACKUP$_READBLOCK                      ; 2342
        00000000G               00              04  FB 000EE  CALLS   #4, FILE_ERROR
                    04          A2              01  88 000F5  BISB2   #1, 4(REC)                             ; 2347
                02              53              57  F2 000F9 9$:  AOBLSS  R7, XBLOCK, 10$                    ; 2320
                                                03  11 000FD  BRB     11$
                                            FF74    31 000FF 10$:  BRW     6$
                                50              68  D0 00102 11$:  MOVL    INPUT_BCB, R0                       ; 2355
                          10    A0      0210    C2  9E 00105      MOVAB   528(R2), 16(R0)
                        FE71    CF              00  FB 0010B      CALLS   #0, SAVE_WRITE                      ; 2356
                          0C    A8              55  C0 00110      ADDL2   BLOCKS, INPUT_BLOCK                 ; 2362
                05        F4    A8              01  E0 00114      BBS     #1, INPUT_FLAGS, 12$                ; 2363
                        0000V   CF              00  FB 00119      CALLS   #0, SAVE_READAHEAD
                                            FEEF    31 0011E 12$:  BRW     1$                                  ; 2274
                                                68  D4 00121 13$:  CLRL    INPUT_BCB                           ; 2366
                                                    04 00123      RET                                         ; 2367

; Routine Size:  292 bytes,    Routine Base:  CODE + 0258
```

```
;   821          2368  1 %SBTTL 'SAVE_READ - read data into save set buffer'
;   822          2369  1 ROUTINE SAVE_READ: NOVALUE=
;   823          2370  1
;   824          2371  1 !++
;   825          2372  1 !
;   826          2373  1 ! FUNCTIONAL DESCRIPTION:
;   827          2374  1 !       This routine gets a buffer if required, formats a data record, and
;   828          2375  1 !       queues it for input.
;   829          2376  1 !
;   830          2377  1 ! INPUT PARAMETERS:
;   831          2378  1 !       NONE
;   832          2379  1 !
;   833          2380  1 ! IMPLICIT INPUTS:
;   834          2381  1 !       NONE
;   835          2382  1 !
;   836          2383  1 ! OUTPUT PARAMETERS:
;   837          2384  1 !       NONE
;   838          2385  1 !
;   839          2386  1 ! IMPLICIT OUTPUTS:
;   840          2387  1 !       NONE
;   841          2388  1 !
;   842          2389  1 ! ROUTINE VALUE:
;   843          2390  1 !       NONE
;   844          2391  1 !
;   845          2392  1 ! SIDE EFFECTS:
;   846          2393  1 !       NONE
;   847          2394  1 !
;   848          2395  1 !--
;   849          2396  1
;   850          2397  2 BEGIN
;   851          2398  2 LOCAL
;   852          2399  2         BLOCKS,                         ! Number of blocks to transfer
;   853          2400  2         REC:            REF BBLOCK,     ! Pointer to record
;   854          2401  2         STATUS;                         ! Status variable
;   855          2402  2
;   856          2403  2
;   857          2404  2 ! Make sure there is something to transfer.  This could fail if, for
;   858          2405  2 ! example, read error recovery reissued enough reads to run off the end.
;   859          2406  2 !
;   860          2407  2 IF .INPUT_BLOCK GTRU .INPUT_MAXBLOCK
;   861          2408  2 THEN
;   862          2409  2     RETURN;
;   863          2410  2
;   864          2411  2
;   865          2412  2 ! Find enough space to hold at least one block.  This can fail because there
;   866          2413  2 ! is a current input buffer on entry to the routine, but there is not enough
;   867          2414  2 ! space in it to hold one block.  If not, release it to the output procedure
;   868          2415  2 ! and get another.
;   869          2416  2 !
;   870          2417  2 WHILE TRUE DO
;   871          2418  3     BEGIN
;   872          2419  3     IF .INPUT_BCB EQL 0 THEN SAVE_GETBUF();
;   873          2420  4     BLOCKS = (.INPUT_BCB[BCB_BUFFER] + .INPUT_BCB[BCB_SIZE] -
;   874          2421  3               .INPUT_BCB[BCB_RECORD] - BRH$C_LENGTH) / 512;
;   875          2422  3     IF .BLOCKS GTR 0 THEN EXITLOOP;
;   876          2423  3     SAVE_WRITE();
;   877          2424  2     END;
```

```
;  878          2425  2
;  879          2426  2
;  880          2427  2 ! Limit the transfer size to the number of blocks remaining.
;  881          2428  2 !
;  882          2429  2 IF .BLOCKS GTR .INPUT_MAXBLOCK + 1 - .INPUT_BLOCK
;  883          2430  2     THEN BLOCKS = .INPUT_MAXBLOCK + 1 - .INPUT_BLOCK;
;  884          2431  2
;  885          2432  2
;  886          2433  2 ! Format the record header.
;  887          2434  2 !
;  888          2435  2 REC = .INPUT_BCB[BCB_RECORD];
;  889          2436  2 REC[BRH$W_RSIZE] = .BLOCKS * 512;
;  890          2437  2 REC[BRH$W_RTYPE] = .INPUT_RTYPE;
;  891          2438  2 REC[BRH$L_FLAGS] = 0;
;  892          2439  2 IF .INPUT_FILECHAR[FCH$V_DIRECTORY] THEN REC[BRH$V_DIRECTORY] = TRUE;
;  893          2440  2 REC[BRH$L_ADDRESS] = .INPUT_BLOCK;
;  894          2441  2 $ASSUME ($BYTEOFFSET (BRH$W_BLOCKFLAGS) + 4, EQL, BRH$K_LENGTH);
;  895          2442  2 (REC[BRH$W_BLOCKFLAGS])<0,32> = 0;
;  896          2443  2
;  897          2444  2
;  898          2445  2 ! Start the read, and make sure it got issued successfully.  Advance the
;  899          2446  2 ! input block counter past the transfer.
;  900          2447  2 !
;  901        P 2448  2 STATUS = $QIO(
;  902        P 2449  2     FUNC=.INPUT_FUNC,
;  903        P 2450  2     CHAN=.INPUT_CHAN,
;  904        P 2451  2     EFN=BCB_S_READ,
;  905        P 2452  2     IOSB=INPUT_BCB[BCB_IOSB],
;  906        P 2453  2     P1=.REC + BRH$C_LENGTH,
;  907        P 2454  2     P2=.BLOCKS * 512,
;  908          2455  2     P3=.INPUT_BLOCK);
;  909          2456  2 IF NOT .STATUS
;  910          2457  2 THEN
;  911          2458  2     FILE_ERROR(
;  912          2459  2         BACKUP$_READERR + STS$K_SEVERE,
;  913          2460  2         .INPUT_FAB,
;  914          2461  2         .STATUS);
;  915          2462  2 INPUT_BLOCK = .INPUT_BLOCK + .BLOCKS;
;  916          2463  2
;  917          2464  2
;  918          2465  2 ! Set up various fields of the BCB, and queue the buffer to the input wait
;  919          2466  2 ! list.  Finally, clear INPUT_BCB to indicate that there is no current buffer.
;  920          2467  2 !
;  921          2468  2 INPUT_BCB[BCB_SUCC_ACT] = 0;
;  922          2469  2 INPUT_BCB[BCB_FAIL_ACT] = 0;
;  923          2470  2 INPUT_BCB[BCB_STATE] = BCB_S_READ;
;  924          2471  2 INSQUE(.INPUT_BCB, .INPUT_QAIT[1]);
;  925          2472  2 INPUT_BCB = 0;
;  926          2473  1 END;


                                                .EXTRN  SYS$QIO

                                003C 00000 SAVE_READ:
                                                .WORD   Save R2,R3,R4,R5                        ; 2369
                     55 00000000'  EF  9E 00002         MOVAB   INPUT_BCB, R5                   ;
```

```
                          10   A5        0C   A5  D1 00009          CMPL    INPUT_BLOCK, INPUT_MAXBLOCK                    ; 2407
                                             01  1B 0000E          BLEQU   1$                                             ;
                                                 04 00010          RET                                                    ;
                                             65  D5 00011 1$:      TSTL    INPUT_BCB                                      ; 2419
                                             05  12 00013          BNEQ    2$                                             ;
                        FC7B   CF                00  FB 00015          CALLS   #0, SAVE_GETBUF                            ;
                               50                65  D0 0001A 2$:      MOVL    INPUT_BCB, R0                              ; 2420
                               51          08    A0  3C 0001D          MOVZWL  8(R0), R1                                  ;
                               51          0C    A0  C0 00021          ADDL2   12(R0), R1                                 ;
                               51          10    A0  C2 00025          SUBL2   16(R0), R1                                 ; 2421
                               51                10  C2 00029          SUBL2   #16, R1                                    ; 2420
                          52               51 00000200  8F  C7 0002C          DIVL3   #512, R1, BLOCKS                           ; 2421
                                             07  14 00034          BGTR    3$                                             ; 2422
                        FE22   CF                00  FB 00036          CALLS   #0, SAVE_WRITE                             ; 2423
                                             D4  11 0003B          BRB     1$                                             ; 2417
                               54          0C    A5  D0 0003D 3$:      MOVL    INPUT_BLOCK, R4                            ; 2429
                          50   10          A5        54  C3 00041          SUBL3   R4, INPUT_MAXBLOCK, R0                     ;
                                                 50  D6 00046          INCL    R0                                         ;
                               50                52  D1 00048          CMPL    BLOCKS, R0                                 ;
                                                 03  15 0004B          BLEQ    4$                                         ;
                               52                50  D0 0004D          MOVL    R0, BLOCKS                                 ; 2430
                               51                65  D0 00050 4$:      MOVL    INPUT_BCB, R1                              ; 2435
                               50          10    A1  D0 00053          MOVL    16(R1), REC                                ;
                          53   52                09  78 00057          ASHL    #9, BLOCKS, R3                             ; 2436
                               60                53  B0 0005B          MOVW    R3, (REC)                                  ;
                             02    A0        ED    A5  9B 0005E          MOVZBW  INPUT_RTYPE, 2(REC)                        ; 2437
                                         04    A0  D4 00063          CLRL    4(REC)                                     ; 2438
                          04   4D    A5              05  E1 00066          BBC     #5, INPUT_FILECHAR+1, 5$                   ; 2439
                               04    A0              02  88 0006B          BISB2   #2, 4(REC)                                 ;
                               08    A0              54  D0 0006F 5$:      MOVL    R4, 8(REC)                                 ; 2440
                                         0C    A0  D4 00073          CLRL    12(REC)                                    ; 2442
                                               7E  7C 00076          CLRQ    -(SP)                                      ; 2455
                                               7E  D4 00078          CLRL    -(SP)                                      ;
                                               18  BB 0007A          PUSHR   #^M<R3,R4>                                 ;
                                         10    A0  9F 0007C          PUSHAB  16(REC)                                    ;
                                               7E  7C 0007F          CLRQ    -(SP)                                      ;
                                         18    A1  9F 00081          PUSHAB  24(R1)                                     ;
                                   7E          EC  A5 9A 00084          MOVZBL  INPUT_FUNC, -(SP)                          ;
                                               F0  A5 DD 00088          PUSHL   INPUT_CHAN                                 ;
                                               01  DD 0008B          PUSHL   #1                                         ;
                        00000000G  00              0C  FB 0008D          CALLS   #12, SYS$QIO                               ;
                                   12              50  E8 00094          BLBS    STATUS, 6$                                 ; 2456
                                               50  DD 00097          PUSHL   STATUS                                     ; 2461
                                               F8  A5 DD 00099          PUSHL   INPUT_FAB                                  ; 2460
                                     00000000G     8F  DD 0009C          PUSHL   #BACKUP$_READERR+4                         ; 2459
                        00000000G  00              03  FB 000A2          CALLS   #3, FILE_ERROR                             ;
                                 0C    A5              52  C0 000A9 6$:      ADDL2   BLOCKS, INPUT_BLOCK                        ; 2462
                                   50              65  D0 000AD          MOVL    INPUT_BCB, R0                              ; 2468
                                         20    A0  7C 000B0          CLRQ    32(R0)                                     ;
                                   0A    A0        01  90 000B3          MOVB    #1, 10(R0)                                 ; 2470
                                 FDEC    D5              60  0E 000B7          INSQUE  (R0), @INPUT_WAIT+4                        ; 2471
                                               65  D4 000BC          CLRL    INPUT_BCB                                  ; 2472
                                                   04 000BE          RET                                                ; 2473

; Routine Size:  191 bytes,    Routine Base:  CODE + 037C
```

```
;  928          2474  1 %SBTTL 'SAVE_READAHEAD - initiate read-ahead'
;  929          2475  1 ROUTINE SAVE_READAHEAD: NOVALUE=
;  930          2476  1
;  931          2477  1 !++
;  932          2478  1 !
;  933          2479  1 ! FUNCTIONAL DESCRIPTION:
;  934          2480  1 !       This routine initiates read-ahead of blocks to be saved.
;  935          2481  1 !
;  936          2482  1 ! INPUT PARAMETERS:
;  937          2483  1 !       NONE
;  938          2484  1 !
;  939          2485  1 ! IMPLICIT INPUTS:
;  940          2486  1 !       NONE
;  941          2487  1 !
;  942          2488  1 ! OUTPUT PARAMETERS:
;  943          2489  1 !       NONE
;  944          2490  1 !
;  945          2491  1 ! IMPLICIT OUTPUTS:
;  946          2492  1 !       NONE
;  947          2493  1 !
;  948          2494  1 ! ROUTINE VALUE:
;  949          2495  1 !       NONE
;  950          2496  1 !
;  951          2497  1 ! SIDE EFFECTS:
;  952          2498  1 !       NONE
;  953          2499  1 !
;  954          2500  1 !--
;  955          2501  1
;  956          2502  2 BEGIN
;  957          2503  2
;  958          2504  2 ! Start reads on half the buffers or until a read has been
;  959          2505  2 ! started on the last block to be transferred.
;  960          2506  2 !
;  961          2507  2 DECR I FROM (.COM_BUFF_COUNT+1)/2 TO 1 DO
;  962          2508  3     BEGIN
;  963          2509  3     IF .INPUT_BLOCK GTRU .INPUT_MAXBLOCK THEN EXITLOOP;
;  964          2510  3     SAVE_READ();
;  965          2511  2     END;
;  966          2512  1 END;


                                        000C 00000 SAVE_READAHEAD:
                                                   .WORD   Save R2,R3                                  ; 2475
                    53 00000000'    EF  9E 00002           MOVAB   COM_BUFF_COUNT, R3                  ;
                    50              63  9A 00009           MOVZBL  COM_BUFF_COUNT, R0                  ; 2507
                                    50  D6 0000C           INCL    R0                                  ;
                52                  02  C7 0000E           DIVL3   #2, R0, R2                          ;
                                    52  D6 00012           INCL    I                                   ;
                                    0C  11 00014           BRB     2$                                  ;
                      67    A3      63  A3  D1 00016 1$:    CMPL    INPUT_BLOCK, INPUT_MAXBLOCK         ; 2509
                                        08  1A 0001B        BGTRU   3$                                  ;
                  FF1F      CF          00  FB 0001D        CALLS   #0, SAVE_READ                       ; 2510
                            F1          52  F5 00022 2$:    SOBGTR  I, 1$                               ; 2507
                                        04  00025 3$:       RET                                         ; 2512
```

; Routine Size: 38 bytes, Routine Base: CODE + 043B

```
;  968          2513  1 %SBTTL 'SAVE_BLOCKS_HANDLER - condition handler for SAVE_BLOCKS'
;  969          2514  1 ROUTINE SAVE_BLOCKS_HANDLER(SIG,MECH)=
;  970          2515  1
;  971          2516  1 !++
;  972          2517  1 !
;  973          2518  1 ! FUNCTIONAL DESCRIPTION:
;  974          2519  1 !       This routine is a handler for routine SAVE_BLOCKS.  It handles getting
;  975          2520  1 !       SAVE_BLOCKS restarted after a reel change.
;  976          2521  1 !
;  977          2522  1 ! INPUT PARAMETERS:
;  978          2523  1 !       SIG             - Standard VMS condition handler parameters
;  979          2524  1 !       MECH            -
;  980          2525  1 !
;  981          2526  1 ! IMPLICIT INPUTS:
;  982          2527  1 !       NONE
;  983          2528  1 !
;  984          2529  1 ! OUTPUT PARAMETERS:
;  985          2530  1 !       NONE
;  986          2531  1 !
;  987          2532  1 ! IMPLICIT OUTPUTS:
;  988          2533  1 !       NONE
;  989          2534  1 !
;  990          2535  1 ! ROUTINE VALUE:
;  991          2536  1 !       SS$_RESIGNAL
;  992          2537  1 !
;  993          2538  1 ! SIDE EFFECTS:
;  994          2539  1 !       NONE
;  995          2540  1 !
;  996          2541  1 !--
;  997          2542  1
;  998          2543  2 BEGIN
;  999          2544  2 MAP
; 1000          2545  2         SIG:            REF BBLOCK,     ! Signal parameters
; 1001          2546  2         MECH:           REF BBLOCK;     ! Mechanism parameters
; 1002          2547  2
; 1003          2548  2
; 1004          2549  2 IF .SIG[CHF$L_SIG_NAME] EQL -1
; 1005          2550  2 THEN
; 1006          2551  3     BEGIN
; 1007          2552  3     MECH[CHF$L_MCH_SAVR0] = FALSE;
; 1008          2553  3     $UNWIND();
; 1009          2554  2     END;
; 1010          2555  2 SS$_RESIGNAL
; 1011          2556  1 END;


                                                        .EXTRN  SYS$UNWIND

                                        0000 00000 SAVE_BLOCKS_HANDLER:
                                                        .WORD   Save nothing                    ; 2514
                        50      04  AC  D0 00002         MOVL    SIG, R0                         ; 2549
            FFFFFFFF    8F      04  A0  D1 00006         CMPL    4(R0), #-1
                                    10  12 0000E         BNEQ    1$                              ;
                        50      08  AC  D0 00010         MOVL    MECH, R0                        ; 2552
                                0C  A0  D4 00014         CLRL    12(R0)                          ;
                                    7E  7C 00017         CLRQ    -(SP)                           ; 2553
```

```
                        00000000G  00              02  FB 00019          CALLS   #2, SYS$UNWIND                          ;
                                   50      0918     8F  3C 00020 1$:     MOVZWL  #2328, R0                              ; 2556
                                                        04 00025          RET                                            ;

; Routine Size:  38 bytes,    Routine Base:  CODE + 0461
```

```
;1013     2557  1 %SBTTL 'SAVE_BLOCKS - save file or volume blocks'
;1014     2558  1 ROUTINE SAVE_BLOCKS=
;1015     2559  1
;1016     2560  1 !++
;1017     2561  1 !
;1018     2562  1 ! FUNCTIONAL DESCRIPTION:
;1019     2563  1 !       This routine is the driver for saving a file or a volume.
;1020     2564  1 !
;1021     2565  1 ! INPUT PARAMETERS:
;1022     2566  1 !       NONE
;1023     2567  1 !
;1024     2568  1 ! IMPLICIT INPUTS:
;1025     2569  1 !       INPUT_BCB       - Points to current buffer, if any.
;1026     2570  1 !       INPUT_BLOCK     - Smallest block number to be saved.
;1027     2571  1 !       INPUT_MAXBLOCK  - Largest block number to be saved.
;1028     2572  1 !       INPUT_RTYPE     - BRH$K_VBN or BRH$K_LBN, as appropriate.
;1029     2573  1 !       INPUT_FUNC      - IO$_READVBLK or IO$_READLBLK, as appropriate.
;1030     2574  1 !       INPUT_CHAN      - Channel number on which input is open.
;1031     2575  1 !
;1032     2576  1 ! OUTPUT PARAMETERS:
;1033     2577  1 !       NONE
;1034     2578  1 !
;1035     2579  1 ! IMPLICIT OUTPUTS:
;1036     2580  1 !       INPUT_BCB       - Points to current buffer.
;1037     2581  1 !
;1038     2582  1 ! ROUTINE VALUE:
;1039     2583  1 !       TRUE, indicating completion.  If a reel-change unwind occurs, the
;1040     2584  1 !       handler forces the routine value to FALSE.
;1041     2585  1 !
;1042     2586  1 ! SIDE EFFECTS:
;1043     2587  1 !       NONE
;1044     2588  1 !
;1045     2589  1 !--
;1046     2590  1
;1047     2591  2 BEGIN
;1048     2592  2 BUILTIN
;1049     2593  2         FP;
;1050     2594  2
;1051     2595  2
;1052     2596  2 ! Establish the handler.
;1053     2597  2 !
;1054     2598  2 .FP = SAVE_BLOCKS_HANDLER;
;1055     2599  2 INPUT_FLAGS[EOV_SAVING] = TRUE;
;1056     2600  2
;1057     2601  2
;1058     2602  2 ! Start reads on the available buffers.
;1059     2603  2 !
;1060     2604  2 SAVE_READAHEAD();
;1061     2605  2
;1062     2606  2
;1063     2607  2 ! Until all blocks have been transferred, wait on the first read, pass it to
;1064     2608  2 ! the output procedure, and start another read.
;1065     2609  2 !
;1066     2610  2 UNTIL .INPUT_BLOCK GTRU .INPUT_MAXBLOCK DO
;1067     2611  3     BEGIN
;1068     2612  3     SAVE_WAIT();
;1069     2613  3     SAVE_READ();
```

```
;  1070          2614  2      END;
;  1071          2615  2
;  1072          2616  2
;  1073          2617  2 ! Until all pending reads have been processed, wait on them, and pass them to
;  1074          2618  2 ! the output procedure.
;  1075          2619  2 !
;  1076          2620  2 UNTIL .INPUT_WAIT[0] EQL INPUT_WAIT[0] DO
;  1077          2621  3     BEGIN
;  1078          2622  3     SAVE_WAIT();
;  1079          2623  2     END;
;  1080          2624  2
;  1081          2625  2
;  1082          2626  2 INPUT_FLAGS[EOV_SAVING] = FALSE;
;  1083          2627  2 TRUE
;  1084          2628  1 END;


                                  0004 00000 SAVE_BLOCKS:
                                             .WORD   Save R2                                 ; 2558
                  52 00000000'    EF  9E 00002         MOVAB   INPUT_FLAGS, R2             ;
                  6D        CE    AF  9E 00009         MOVAB   SAVE_BLOCKS_HANDLER, (FP)       ; 2598
                  62              04  88 0000D         BISB2   #4, INPUT_FLAGS                 ; 2599
                A0    AF          00  FB 00010         CALLS   #0, SAVE_READAHEAD              ; 2604
                1C    A2        18    A2  D1 00014 1$:   CMPL   INPUT_BLOCK, INPUT_MAXBLOCK      ; 2610
                                  0C  1A 00019         BGTRU   2$                          ;
              FDB1    CF            00  FB 0001B         CALLS   #0, SAVE_WAIT                   ; 2612
              FED0    CF            00  FB 00020         CALLS   #0, SAVE_READ                   ; 2613
                                  ED  11 00025         BRB     1$                              ; 2610
                  50      FDF4    C2  9E 00027 2$:     MOVAB   INPUT_WAIT, R0                  ; 2620
                  50      FDF4    C2  D1 0002C         CMPL    INPUT_WAIT, R0              ;
                                  07  13 00031         BEQL    3$                          ;
              FD99    CF            00  FB 00033         CALLS   #0, SAVE_WAIT                   ; 2622
                                  ED  11 00038         BRB     2$                              ; 2620
                  62              04  8A 0003A 3$:     BICB2   #4, INPUT_FLAGS                 ; 2626
                  50              01  D0 0003D         MOVL    #1, R0                          ; 2628
                                      04 00040         RET                                 ;

; Routine Size:  65 bytes,    Routine Base:  CODE + 0487
```

```
;1086        2629  1 %SBTTL 'BACKUP_SUMMARY - format volume summary record'
;1087        2630  1 ROUTINE BACKUP_SUMMARY: NOVALUE=
;1088        2631  1
;1089        2632  1 !++
;1090        2633  1 !
;1091        2634  1 ! FUNCTIONAL DESCRIPTION:
;1092        2635  1 !       This routine generates the backup summary record.
;1093        2636  1 !
;1094        2637  1 ! INPUT PARAMETERS:
;1095        2638  1 !       NONE
;1096        2639  1 !
;1097        2640  1 ! IMPLICIT INPUTS:
;1098        2641  1 !       NONE
;1099        2642  1 !
;1100        2643  1 ! OUTPUT PARAMETERS:
;1101        2644  1 !       NONE
;1102        2645  1 !
;1103        2646  1 ! IMPLICIT OUTPUTS:
;1104        2647  1 !       NONE
;1105        2648  1 !
;1106        2649  1 ! ROUTINE VALUE:
;1107        2650  1 !       NONE
;1108        2651  1 !
;1109        2652  1 ! SIDE EFFECTS:
;1110        2653  1 !       NONE
;1111        2654  1 !
;1112        2655  1 !--
;1113        2656  1
;1114        2657  2 BEGIN
;1115        2658  2 LOCAL
;1116        2659  2         L1,
;1117        2660  2         L2,
;1118        2661  2         P;
;1119        2662  2 LITERAL
;1120        2663  2     FIXED_SIZE =
;1121        2664  2         BRH$C_LENGTH +          ! record header
;1122        2665  2         2 +                     ! structure level
;1123        2666  2         12*4 +                  ! 12 attribute size/type longwords
;1124        2667  2                                 !  1. save set name (variable)
;1125        2668  2                                 !  2. command line (variable)
;1126        2669  2         12 +                    !  3. user name
;1127        2670  2         BSA$S_USERUIC +         !  4. UIC of user
;1128        2671  2         BSA$S_DATE +            !  5. date backup was done
;1129        2672  2         BSA$S_OPSYS +           !  6. operating system
;1130        2673  2         BSA$S_SYSVER +          !  7. operating system version
;1131        2674  2         BSA$S_SIR +             !  8. CPU system ID register
;1132        2675  2         %CHARCOUNT(BACKUP$VERSION) + !
;1133        2676  2                                 !  9. version number of BACKUP
;1134        2677  2         BSA$S_BLOCKSIZE +       ! 10. block size of save set
;1135        2678  2         BSA$S_XORSIZE +         ! 11. size of each XOR group
;1136        2679  2         BSA$S_BUFFERS;          ! 12. number of buffers
;1137        2680  2
;1138        2681  2
;1139        2682  2 L1 =
;1140        2683  2     FIXED_SIZE +
;1141        2684  2     .COM_$SNAME[DSC$W_LENGTH] +
;1142        2685  2     .BBLOCK[QUAL[QUAL_CMD_DESC], DSC$W_LENGTH];
```

```
;1143          2686  2
;1144          2687  2
;1145          2688  2 IF .QUAL[QUAL_COMM]
;1146          2689  2 THEN
;1147          2690  2     L1 = .L1 +
;1148          2691  2          4 +                         ! 1 attribute size/type longword
;1149          2692  2          .BBLOCK[QUAL[QUAL_COMM_DESC], DSC$W_LENGTH];
;1150          2693  2
;1151          2694  2
;1152          2695  2 IF .JPI_NODE_DESC[DSC$W_LENGTH] NEQ 0
;1153          2696  2 THEN
;1154          2697  2     L1 = .L1 +
;1155          2698  2          4 +
;1156          2699  2          .JPI_NODE_DESC[DSC$W_LENGTH];
;1157          2700  2
;1158          2701  2
;1159          2702  2 IF .BBLOCK[RWSV_SAVE_QUAL[QUAL_DVI_DESC], DSC$W_LENGTH] NEQ 0
;1160          2703  2 THEN
;1161          2704  2     L1 = .L1 +
;1162          2705  2          4 +
;1163          2706  2          .BBLOCK[RWSV_SAVE_QUAL[QUAL_DVI_DESC], DSC$W_LENGTH] + 1;
;1164          2707  2
;1165          2708  2
;1166          2709  2 IF .QUAL[QUAL_IMAG]
;1167          2710  2 THEN
;1168          2711  2     L1 = .L1 +
;1169          2712  2          2*4 +                       ! 2 attribute size/type longwords
;1170          2713  2          BSA$S_VOLSETNAM +           ! 16. volume set name
;1171          2714  2          BSA$S_NVOLS;                ! 17. number of volumes in set
;1172          2715  2
;1173          2716  2
;1174          2717  2 IF .COM_FLAGS[COM_STANDALONE]
;1175          2718  2 THEN
;1176          2719  2     L1 = .L1 + %CHARCOUNT(' (standalone)');
;1177          2720  2
;1178          2721  2 IF .QUAL[QUAL_SS_ENCRYP]
;1179          2722  2 THEN
;1180          2723  2          L1 = .L1 +
;1181          2724  2               4+                     ! 1 attribute/size longword
;1182          2725  2               BSA$S_CRYPDATKEY ; ! size of cryp key data
;1183          2726  2
;1184          2727  2 L2 = (.L1 + 15) AND NOT 15;
;1185          2728  2 P = MAKE_SPACE(.L2);
;1186          2729  2
;1187          2730  2
;1188          2731  2 STORE_(2, .L2 - BRH$C_LENGTH);
;1189          2732  2 STORE_(2, BRH$K_SUMMARY);
;1190          2733  2 STORE_(4, 0);
;1191          2734  2 STORE_(4, 0);
;1192          2735  2 STORE_(4, 0);
;1193          2736  2 STORE_(2, BBH$K_LEVEL1);
;1194          2737  2
;1195          2738  2
;1196        P 2739  2 ATV_('SSNAME',
;1197        P 2740  2     .COM_SSNAME[DSC$W_LENGTH],
;1198          2741  2     .COM_SSNAME[DSC$A_POINTER]);
;1199          2742  2
```

```
;  1200          2743  2
;  1201        P 2744  2 ATV_('COMMAND',
;  1202        P 2745  2     .BBLOCK[QUAL[QUAL_CMD_DESC], DSC$W_LENGTH],
;  1203          2746  2     .BBLOCK[QUAL[QUAL_CMD_DESC], DSC$A_POINTER]);
;  1204          2747  2
;  1205          2748  2
;  1206          2749  2 IF .QUAL[QUAL_COMM]
;  1207          2750  2 THEN
;  1208        P 2751  2     ATV_('COMMENT',
;  1209        P 2752  2         .BBLOCK[QUAL[QUAL_COMM_DESC], DSC$W_LENGTH],
;  1210          2753  2         .BBLOCK[QUAL[QUAL_COMM_DESC], DSC$A_POINTER]);
;  1211          2754  2
;  1212          2755  2
;  1213          2756  2 ATV_('USERNAME', 12, JPI_USERNAME);
;  1214          2757  2 ATT_('USERUIC',  .JPI_UIC);
;  1215          2758  2 ATT_('DATE',     JPI_DATE);
;  1216          2759  2 ATT_('OPSYS',    BACKUP$K_OPSYS);
;  1217          2760  2 ATT_('SYSVER',   .SYI_VERSION);
;  1218          2761  2
;  1219          2762  2
;  1220          2763  2 IF .JPI_NODE_DESC[DSC$W_LENGTH] NEQ 0
;  1221          2764  2 THEN
;  1222        P 2765  2     ATV_('NODENAME',
;  1223        P 2766  2     .JPI_NODE_DESC[DSC$W_LENGTH],
;  1224          2767  2     .JPI_NODE_DESC[DSC$A_POINTER]);
;  1225          2768  2
;  1226          2769  2
;  1227          2770  2 ATT_('SIR',      .SYI_SID);
;  1228          2771  2
;  1229          2772  2
;  1230          2773  2 IF .BBLOCK[RWSV_SAVE_QUAL[QUAL_DVI_DESC], DSC$W_LENGTH] NEQ 0
;  1231          2774  2 THEN
;  1232        P 2775  2     DVI_('DRIVEID',
;  1233        P 2776  2         .BBLOCK[RWSV_SAVE_QUAL[QUAL_DVI_DESC], DSC$W_LENGTH],
;  1234          2777  2         .BBLOCK[RWSV_SAVE_QUAL[QUAL_DVI_DESC], DSC$A_POINTER]);
;  1235          2778  2
;  1236          2779  2
;  1237          2780  2 IF .COM_FLAGS[COM_STANDALONE]
;  1238          2781  2 THEN
;  1239        P 2782  2     ATV_(
;  1240        P 2783  2         'BACKVER',
;  1241        P 2784  2         %CHARCOUNT(BACKUP$VERSION, ' (standalone)'),
;  1242          2785  3         VERSION_STRING)
;  1243          2786  2 ELSE
;  1244        P 2787  2     ATV_(
;  1245        P 2788  2         'BACKVER',
;  1246        P 2789  2         %CHARCOUNT(BACKUP$VERSION),
;  1247          2790  2         VERSION_STRING);
;  1248          2791  2
;  1249          2792  2
;  1250          2793  2 ATT_('BLOCKSIZE',.QUAL[QUAL_BLOC_VALUE]);
;  1251          2794  2 ATT_('XORSIZE',  .QUAL[QUAL_GROU_VALUE]);
;  1252          2795  2 ATT_('BUFFERS',  .QUAL[QUAL_BUFF_VALUE]);
;  1253          2796  2
;  1254          2797  2
;  1255          2798  2 IF .QUAL[QUAL_IMAG]
;  1256          2799  2 THEN
```

```
;  1257          2800  3     BEGIN
;  1258          2801  3     ATT_('VOLSETNAM',   COM_I_STRUCNAME);
;  1259          2802  3     ATT_('NVOLS',       .COM_I_SETCOUNT);
;  1260          2803  2     END;
;  1261          2804  2
;  1262          2805  2 ! Move saveset encryption data to summary record
;  1263          2806  2 !
;  1264          2807  2 IF .QUAL[QUAL_SS_ENCRYP]
;  1265          2808  2 THEN
;  1266          2809  2     ATV_('CRYPDATKEY',  BSA$S_CRYPDATKEY, CRYP_DATA_CODE);
;  1267          2810  2
;  1268          2811  2 CH$FILL(0, .L2 - .L1, .P);
;  1269          2812  1 END;


                                        03FC 00000 BACKUP_SUMMARY:
                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                    ; 2630
                                59      FB32   CF  9E 00002         MOVAB   VERSION_STRING, R9
                                58 00000000'   EF  9E 00007         MOVAB   QUAL+8, R8
                                50        68   A8  3C 0000E         MOVZWL  COM_SSNAME, R0                 ; 2685
                                51        20   A8  3C 00012         MOVZWL  QUAL+40, R1
                                50             51  C0 00016         ADDL2   R1, R0
                                56        70   A0  9E 00019         MOVAB   112(R0), L1                    ; 2684
                     09         68             06  E1 0001D         BBC     #6, QUAL+8, 1$                 ; 2688
                                50        10   A8  3C 00021         MOVZWL  QUAL+24, R0                    ; 2692
                                56        04 A046  9E 00025         MOVAB   4(R0)[L1], L1                  ; 2691
                                50      FEE4   C8  3C 0002A 1$:     MOVZWL  JPI_NODE_DESC, R0              ; 2695
                                               05  13 0002F         BEQL    2$
                                56        04 A046  9E 00031         MOVAB   4(R0)[L1], L1                  ; 2698
                                50        A0   A8  D0 00036 2$:     MOVL    RWSV_SAVE_QUAL, R0             ; 2702
                                50        18   A0  3C 0003A         MOVZWL  24(R0), R0
                                               05  13 0003E         BEQL    3$
                                56        05 A046  9E 00040         MOVAB   5(R0)[L1], L1                  ; 2706
                   03     02   A8             03  E1 00045 3$:     BBC     #3, QUAL+10, 4$                ; 2709
                                56             16  C0 0004A         ADDL2   #22, L1                        ; 2713
                   03     72   A8             01  E1 0004D 4$:     BBC     #1, COM_FLAGS, 5$              ; 2717
                                56             0D  C0 00052         ADDL2   #13, L1                        ; 2719
                   03     06   A8             04  E1 00055 5$:     BBC     #4, QUAL+14, 6$                ; 2721
                                56             1C  C0 0005A         ADDL2   #28, L1                        ; 2724
                                50        0F   A6  9E 0005D 6$:     MOVAB   15(R6), R0                     ; 2727
                   57           50             0F  CB 00061         BICL3   #15, R0, L2
                                               57  DD 00065         PUSHL   L2                             ; 2728
                         FC6C   CF             01  FB 00067         CALLS   #1, MAKE_SPACE
                   80           57             10  A3 0006C         SUBW3   #16, L2, (P)+                  ; 2731
                                80             01  B0 00070         MOVW    #1, (P)+                       ; 2732
                                               80  7C 00073         CLRQ    (P)+                           ; 2733
                                               80  D4 00075         CLRL    (P)+                           ; 2735
                                80      0101   8F  B0 00077         MOVW    #257, (P)+                     ; 2736
                                80        68   A8  B0 0007C         MOVW    COM_SSNAME, (P)+               ; 2741
                                80             01  B0 00080         MOVW    #1, (P)+
                   60     6C   B8        68   A8  28 00083         MOVC3   COM_SSNAME, @COM_SSNAME+4, (P)
                                50             53  D0 00089         MOVL    R3, P
                                80        20   A8  B0 0008C         MOVW    QUAL+40, (P)+                  ; 2746
                                80             02  B0 00090         MOVW    #2, (P)+
```

```
                60      24   B8      20   A8  28 00093          MOVC3   QUAL+40, @QUAL+44, (P)                          ;
                50           53  D0 00099          MOVL    R3, P                                                   ;
                10           68      06  E1 0009C          BBC     #6, QUAL+8, 7$                                  ; 2749
                             80      10  A8  B0 000A0          MOVW    QUAL+24, (P)+                                   ; 2753
                             80          03  B0 000A4          MOVW    #3, (P)+                                        ;
                60      14   B8      10  A8  28 000A7          MOVC3   QUAL+24, @QUAL+28, (P)                          ;
                50           53  D0 000AD          MOVL    R3, P                                                   ;
                             80 0004000C 8F  D0 000B0 7$:       MOVL    #262156, (P)+                                   ; 2756
                60      FED0 C8          0C  28 000B7          MOVC3   #12, JPI_USERNAME, (P)                          ;
                50           53  D0 000BD          MOVL    R3, P                                                   ;
                             80 00050004 8F  D0 000C0          MOVL    #327684, (P)+                                   ; 2757
                             80      FECC C8  D0 000C7          MOVL    JPI_UIC, (P)+                                   ;
                             80 00060008 8F  D0 000CC          MOVL    #393224, (P)+                                   ; 2758
                             80      FEDC C8  7D 000D3          MOVQ    JPI_DATE, (P)+                                  ;
                             80 00070002 8F  D0 000D8          MOVL    #458754, (P)+                                   ; 2759
                             80      0400 8F  B0 000DF          MOVW    #1024, (P)+                                     ;
                             80 00080004 8F  D0 000E4          MOVL    #524292, (P)+                                   ; 2760
                             80      FEF4 C8  D0 000EB          MOVL    SYI_VERSION, (P)+                               ;
                             51      FEE4 C8  3C 000F0          MOVZWL  JPI_NODE_DESC, R1                               ; 2763
                                          0F  13 000F5          BEQL    8$                                              ;
                             80          51  B0 000F7          MOVW    R1, (P)+                                        ; 2767
                             80          09  B0 000FA          MOVW    #9, (P)+                                        ;
                60      FEE8 D8          51  28 000FD          MOVC3   R1, @JPI_NODE_DESC+4, (P)                       ;
                50           53  D0 00103          MOVL    R3, P                                                   ;
                             80 000A0004 8F  D0 00106 8$:       MOVL    #655364, (P)+                                   ; 2770
                             80      FEF8 C8  D0 0010D          MOVL    SYI_SID, (P)+                                   ;
                             51        A0 A8  D0 00112          MOVL    RWSD_SAVE_QUAL, R1                              ; 2773
                             51           18  C0 00116          ADDL2   #24, R1                                         ;
                                          61  B5 00119          TSTW    (R1)                                            ;
                                          12  13 0011B          BEQL    9$                                              ;
            80                61          01  A1 0011D          ADDW3   #1, (R1), (P)+                                  ; 2777
                              80          0B  B0 00121          MOVW    #11, (P)+                                       ;
            60          04    B1          61  28 00124          MOVC3   (R1), @4(R1), (P)                               ;
                              50          53  D0 00129          MOVL    R3, P                                           ;
                              80          3A  90 0012C          MOVB    #58, (P)+                                       ;
            10          72    A8          01  E1 0012F 9$:       BBC     #1, COM_FLAGS, 10$                              ; 2780
                              80 000C0011 8F  D0 00134          MOVL    #786449, (P)+                                   ; 2785
            60                69          11  28 0013B          MOVC3   #17, VERSION_STRING, (P)                        ;
                              50          53  D0 0013F          MOVL    R3, P                                           ;
                                          0A  11 00142          BRB     11$                                             ;
                              80 000C0004 8F  D0 00144 10$:      MOVL    #786436, (P)+                                   ; 2790
                              80          69  D0 0014B          MOVL    VERSION_STRING, (P)+                            ;
                              80 000D0004 8F  D0 0014E 11$:      MOVL    #851972, (P)+                                   ; 2793
                              80        40 A8  3C 00155          MOVZWL  QUAL+72, (P)+                                   ;
                              80 000E0002 8F  D0 00159          MOVL    #917506, (P)+                                   ; 2794
                              80        46 A8  9B 00160          MOVZBW  QUAL+78, (P)+                                   ;
                              80 000F0002 8F  D0 00164          MOVL    #983042, (P)+                                   ; 2795
                              80        44 A8  9B 0016B          MOVZBW  QUAL+76, (P)+                                   ;
            1B          02    A8          03  E1 0016F          BBC     #3, QUAL+10, 12$                                ; 2798
                              80 0010000C 8F  D0 00174          MOVL    #1048588, (P)+                                  ; 2801
                              80        78 A8  7D 0017B          MOVQ    COM_I_STRUCNAME, (P)+                           ;
                              80      0080 C8  D0 0017F          MOVL    COM_I_STRUCNAME+8, (P)+                         ;
                              80 00110002 8F  D0 00184          MOVL    #1114114, (P)+                                  ; 2802
                              80        76 A8  9B 0018B          MOVZBW  COM_I_SETCOUNT, (P)+                            ;
            10          08    A8          04  E1 0018F 12$:      BBC     #4, QUAL+14, 13$                                ; 2807
                              80 00510018 8F  D0 00194          MOVL    #5308440, (P)+                                  ; 2809
            60        06E0    C8          18  28 0019B          MOVC3   #24, CRYP_DATA_CODE, (P)                        ;
```

```
                                50              53  D0 001A1           MOVL    R3, P                                   ;
                                57              56  C2 001A4 13$:      SUBL2   L1, R7                                  ; 2811
          57            00      6E              00  2C 001A7           MOVC5   #0, (SP), #0, R7, (P)                   ;
                                                60     001AC                                                           ;
                                                   04 001AD           RET                                             ; 2812

; Routine Size:  430 bytes,    Routine Base:  CODE + 04C8
```

```
;  1271          2813  1 %SBTTL 'FROM_ODS1_DATE - convert ODS-1 date to 64 bit'
;  1272          2814  1 GLOBAL ROUTINE FROM_ODS1_DATE(ADDRESS,ATT_BUFFER): NOVALUE=
;  1273          2815  1
;  1274          2816  1 !++
;  1275          2817  1 !
;  1276          2818  1 ! FUNCTIONAL DESCRIPTION:
;  1277          2819  1 !       This routine converts a date in ODS-1 format to 64-bit format.
;  1278          2820  1 !
;  1279          2821  1 ! INPUT PARAMETERS:
;  1280          2822  1 !       ADDRESS         - Pointer to input date string.
;  1281          2823  1 !       ATT_BUFFER      - Pointer to output quadword.
;  1282          2824  1 !
;  1283          2825  1 ! IMPLICIT INPUTS:
;  1284          2826  1 !       NONE
;  1285          2827  1 !
;  1286          2828  1 ! OUTPUT PARAMETERS:
;  1287          2829  1 !       NONE
;  1288          2830  1 !
;  1289          2831  1 ! IMPLICIT OUTPUTS:
;  1290          2832  1 !       NONE
;  1291          2833  1 !
;  1292          2834  1 ! ROUTINE VALUE:
;  1293          2835  1 !       NONE
;  1294          2836  1 !
;  1295          2837  1 ! SIDE EFFECTS:
;  1296          2838  1 !       NONE
;  1297          2839  1 !
;  1298          2840  1 !--
;  1299          2841  1
;  1300          2842  2 BEGIN
;  1301          2843  2 MAP
;  1302          2844  2         ATT_BUFFER:     REF VECTOR;             ! Pointer to output buffer
;  1303          2845  2 LITERAL
;  1304          2846  2         DATLEN=         23;
;  1305          2847  2 LOCAL
;  1306          2848  2         DATBUF:         VECTOR[DATLEN,BYTE],    ! Buffer for $BINTIM
;  1307          2849  2         DATDESC:        VECTOR[2];              ! Descriptor for buffer
;  1308          2850  2
;  1309          2851  2
;  1310          2852  2 IF .(.ADDRESS)<0,8> EQL 0
;  1311          2853  2 THEN
;  1312          2854  3     BEGIN
;  1313          2855  3     ATT_BUFFER[0] = ATT_BUFFER[1] = 0;
;  1314          2856  3     RETURN;
;  1315          2857  2     END;
;  1316          2858  2
;  1317          2859  2
;  1318          2860  2 (DATBUF+00)<0,16> = .(.ADDRESS);
;  1319          2861  2 (DATBUF+02)<0,8>  = '-';
;  1320          2862  2 (DATBUF+03)<0,24> = .(.ADDRESS+2);
;  1321          2863  2 (DATBUF+06)<0,24> = '-19';
;  1322          2864  2 (DATBUF+09)<0,16> = .(.ADDRESS+5);
;  1323          2865  2 (DATBUF+11)<0,8>  = ' ';
;  1324          2866  2 IF .(.ADDRESS+7)<0,8> NEQ 0
;  1325          2867  2 THEN
;  1326          2868  3     BEGIN
;  1327          2869  3     (DATBUF+12)<0,16> = .(.ADDRESS+7);
```

```
;  1328        2870  3      (DATBUF+14)<0,8>  = ':';
;  1329        2871  3      (DATBUF+15)<0,16> = .(.ADDRESS+9);
;  1330        2872  3      (DATBUF+17)<0,8>  = ':';
;  1331        2873  3      (DATBUF+18)<0,16> = .(.ADDRESS+11);
;  1332        2874  3      END
;  1333        2875  2 ELSE
;  1334        2876  3      BEGIN
;  1335        2877  3      (DATBUF+12)<0,32> = '00:0';
;  1336        2878  3      (DATBUF+16)<0,32> = '0:00';
;  1337        2879  2      END;
;  1338        2880  2 (DATBUF+20)<0,24> = '.00';
;  1339        2881  2
;  1340        2882  2
;  1341        2883  2 DATDESC[0] = DATLEN;
;  1342        2884  2 DATDESC[1] = DATBUF;
;  1343        2885  2 $BINTIM(TIMBUF=DATDESC, TIMADR=.ATT_BUFFER);
;  1344        2886  1 END;


                                                                .EXTRN  SYS$BINTIM

                                        0000 00000              .ENTRY  FROM_ODS1_DATE, Save nothing                    ; 2814
                                    5E  20  C2 00002            SUBL2   #32, SP                                         ;
                                    51  04  AC  D0 00005        MOVL    ADDRESS, R1                                     ; 2852
                                            61  95 00009        TSTB    (R1)                                            ;
                                            07  12 0000B        BNEQ    1$                                              ;
                                    50      08  AC  D0 0000D    MOVL    ATT_BUFFER, R0                                  ; 2855
                                            60  7C 00011        CLRQ    (R0)                                            ;
                                                04 00013        RET                                                     ; 2854
                                        08  AE      61  B0 00014 1$:     MOVW    (R1), DATBUF                            ; 2860
                                        0A  AE      2D  90 00018        MOVB    #45, DATBUF+2                           ; 2861
                0B  AE          18          00  02  A1  F0 0001C        INSV    2(R1), #0, #24, DATBUF+3                ; 2862
                0E  AE          18          00 0039312D 8F  FC 00023    INSV    #3748141, #0, #24, DATBUF+6             ; 2863
                                        11  AE      05  A1  B0 0002D    MOVW    5(R1), DATBUF+9                         ; 2864
                                        13  AE          20  90 00032    MOVB    #32, DATBUF+11                          ; 2865
                                                07  A1  95 00036        TSTB    7(R1)                                   ; 2866
                                                    19  13 00039        BEQL    2$                                      ;
                                        14  AE      07  A1  B0 0003B    MOVW    7(R1), DATBUF+12                        ; 2869
                                        16  AE          3A  90 00040    MOVB    #58, DATBUF+14                          ; 2870
                                        17  AE      09  A1  B0 00044    MOVW    9(R1), DATBUF+15                        ; 2871
                                        19  AE          3A  90 00049    MOVB    #58, DATBUF+17                          ; 2872
                                        1A  AE      0B  A1  B0 0004D    MOVW    11(R1), DATBUF+18                       ; 2873
                                                    10  11 00052        BRB     3$                                      ; 2866
                                        14  AE 303A3030 8F  D0 00054 2$:     MOVL    #809119792, DATBUF+12               ; 2877
                                        18  AE 30303A30 8F  D0 0005C        MOVL    #808466992, DATBUF+16               ; 2878
                0C  AE          18          00 0030302E 8F  F0 00064 3$:     INSV    #3158062, #0, #24, DATBUF+20        ; 2880
                                            6E          17  D0 0006E        MOVL    #23, DATDESC                        ; 2883
                                        04  AE      08  AE  9E 00071        MOVAB   DATBUF, DATDESC+4                   ; 2884
                                                08  AC  DD 00076        PUSHL   ATT_BUFFER                          ; 2885
                                                04  AE  9F 00079        PUSHAB  DATDESC                             ;
                                00000000G   00          02  FB 0007C    CALLS   #2, SYS$BINTIM                      ;
                                                        04 00083        RET                                         ; 2886

; Routine Size:  132 bytes,    Routine Base:  CODE + 0676
```

```
;  1346     2887  1 %SBTTL 'ALLOC_PLC_BLOCK - allocate placement data'
;  1347     2888  1 ROUTINE ALLOC_PLC_BLOCK(TYPE,SIZE,DATA): NOVALUE=
;  1348     2889  1
;  1349     2890  1 !++
;  1350     2891  1 !
;  1351     2892  1 ! FUNCTIONAL DESCRIPTION:
;  1352     2893  1 !       This routine allocates and links in a block of placement data.
;  1353     2894  1 !
;  1354     2895  1 ! INPUT PARAMETERS:
;  1355     2896  1 !       TYPE            - Type code for placement block.
;  1356     2897  1 !       SIZE            - Size of placement data.
;  1357     2898  1 !       DATA            - Pointer to data for placement block.
;  1358     2899  1 !
;  1359     2900  1 ! IMPLICIT INPUTS:
;  1360     2901  1 !       NONE
;  1361     2902  1 !
;  1362     2903  1 ! OUTPUT PARAMETERS
;  1363     2904  1 !       NONE
;  1364     2905  1 !
;  1365     2906  1 ! IMPLICIT OUTPUTS:
;  1366     2907  1 !       NONE
;  1367     2908  1 !
;  1368     2909  1 ! ROUTINE VALUE:
;  1369     2910  1 !       NONE
;  1370     2911  1 !
;  1371     2912  1 ! SIDE EFFECTS:
;  1372     2913  1 !       New block allocated and linked to listhead.
;  1373     2914  1 !
;  1374     2915  1 !--
;  1375     2916  1
;  1376     2917  2 BEGIN
;  1377     2918  2 LOCAL
;  1378     2919  2         P:              REF BBLOCK;     ! Pointer to allocated space
;  1379     2920  2
;  1380     2921  2
;  1381     2922  2 ! Allocate the new block.
;  1382     2923  2 !
;  1383     2924  2 P = GET_VM(PLC_S_HDR + .SIZE);
;  1384     2925  2
;  1385     2926  2
;  1386     2927  2 ! Link new block to tail of list and initialize it.
;  1387     2928  2 !
;  1388     2929  2 INSQUE(.P, .INPUT_PLACEMENT[1]);
;  1389     2930  2 P[PLC_TYPE] = .TYPE;
;  1390     2931  2 P[PLC_SIZE] = PLC_S_HDR + .SIZE;
;  1391     2932  2 CH$MOVE(.SIZE, .DATA, P[PLC_DATA]);
;  1392     2933  2
;  1393     2934  2
;  1394     2935  2 ! Count size of new block.
;  1395     2936  2 !
;  1396     2937  2 INPUT_PLACE_LEN = .INPUT_PLACE_LEN + .SIZE + 1;
;  1397     2938  1 END;
```

```
                                          007C 00000 ALLOC_PLC_BLOCK:
                                                               .WORD   Save R2,R3,R4,R5,R6                          ; 2888
                          56 00000000'  EF  9E 00002           MOVAB   INPUT_PLACE_LEN, R6                          ;
        7E          08    AC            0A  C1 00009           ADDL3   #10, SIZE, -(SP)                             ; 2924
             00000000G    00            01  FB 0000E           CALLS   #1, GET_VM                                   ;
                    F4    B6            60  0E 00015           INSQUE  (P), @INPUT_PLACEMENT+4                      ; 2929
                    08    A0      04    AC  90 00019           MOVB    TYPE, 8(P)                                   ; 2930
   09   A0          08    AC            0A  81 0001E           ADDB3   #10, SIZE, 9(P)                              ; 2931
   0A   A0          0C    BC      08    AC  28 00024           MOVC3   SIZE, @DATA, 10(P)                           ; 2932
                          50            66  3C 0002B           MOVZWL  INPUT_PLACE_LEN, R0                          ; 2937
                          50      08    AC  C0 0002E           ADDL2   SIZE, R0                                     ;
        66                50            01  A1 00032           ADDW3   #1, R0, INPUT_PLACE_LEN                      ;
                                            04 00036           RET                                                  ; 2938

; Routine Size:  55 bytes,    Routine Base:  CODE + 06FA
```

```
;  1399          2939  1 %SBTTL 'ALLOC_VBN_BLOCK - allocate VBN data'
;  1400          2940  1 ROUTINE ALLOC_VBN_BLOCK(VBN,COUNT): NOVALUE=
;  1401          2941  1
;  1402          2942  1 !++
;  1403          2943  1 !
;  1404          2944  1 ! FUNCTIONAL DESCRIPTION:
;  1405          2945  1 !       This routine allocates and links in a block of VBN data.
;  1406          2946  1 !
;  1407          2947  1 ! INPUT PARAMETERS:
;  1408          2948  1 !       VBN             - Starting VBN.
;  1409          2949  1 !       COUNT           - Block count.
;  1410          2950  1 !
;  1411          2951  1 ! IMPLICIT INPUTS:
;  1412          2952  1 !       NONE
;  1413          2953  1 !
;  1414          2954  1 ! OUTPUT PARAMETERS:
;  1415          2955  1 !       NONE
;  1416          2956  1 !
;  1417          2957  1 ! IMPLICIT OUTPUTS:
;  1418          2958  1 !       NONE
;  1419          2959  1 !
;  1420          2960  1 ! ROUTINE VALUE:
;  1421          2961  1 !       NONE
;  1422          2962  1 !
;  1423          2963  1 ! SIDE EFFECTS:
;  1424          2964  1 !       New block allocated and linked to listhead.
;  1425          2965  1 !
;  1426          2966  1 !--
;  1427          2967  1
;  1428          2968  2 BEGIN
;  1429          2969  2 LOCAL
;  1430          2970  2         P:              REF BBLOCK,     ! Pointer to allocated space
;  1431          2971  2         HIGH_VBN;                       ! Highest VBN in range
;  1432          2972  2
;  1433          2973  2
;  1434          2974  2 INPUT_FLAGS[INPUT_ON_RVN] = TRUE;
;  1435          2975  2 HIGH_VBN = MINU(.VBN + .COUNT - 1, .INPUT_MAXBLOCK);
;  1436          2976  2 IF .HIGH_VBN GEQU .VBN
;  1437          2977  2 THEN
;  1438          2978  3     BEGIN
;  1439          2979  3
;  1440          2980  3     ! Allocate the new block.
;  1441          2981  3     !
;  1442          2982  3     P = GET_VM(VBN_S_ENTRY);
;  1443          2983  3
;  1444          2984  3
;  1445          2985  3     ! Link new block to tail of list and initialize it.
;  1446          2986  3     !
;  1447          2987  3     INSQUE(.P, .INPUT_VBN_LIST[1]);
;  1448          2988  3     P[PLC_SIZE] = VBN_S_ENTRY;
;  1449          2989  3     P[VBN_FIRST] = .VBN;
;  1450          2990  3     P[VBN_LAST] = .HIGH_VBN;
;  1451          2991  2     END;
;  1452          2992  1 END;
```

```
                                          000C 00000 ALLOC_VBN_BLOCK:
                                                            .WORD    Save R2,R3                            ; 2940
                    53 00000000'    EF  9E 00002            MOVAB    INPUT_MAXBLOCK, R3                    ;
              E4    A3              08  88 00009            BISB2    #8, INPUT_FLAGS                       ; 2974
       50     04    AC        08    AC  C1 0000D            ADDL3    COUNT, VBN, R0                        ; 2975
                                    50  D7 00013            DECL     R0                                    ;
                    63              50  D1 00015            CMPL     R0, INPUT_MAXBLOCK                    ;
                                    03  1B 00018            BLEQU    1$                                    ;
                    50              63  D0 0001A            MOVL     INPUT_MAXBLOCK, R0                    ;
                    52              50  D0 0001D 1$:        MOVL     R0, HIGH_VBN                          ;
              04    AC              52  D1 00020            CMPL     HIGH_VBN, VBN                         ; 2976
                                    1A  1F 00024            BLSSU    2$                                    ;
                                    12  DD 00026            PUSHL    #18                                   ; 2982
         00000000G  00              01  FB 00028            CALLS    #1, GET_VM                            ;
              70    B3              60  0E 0002F            INSQUE   (P), @INPUT_VBN_LIST+4                ; 2987
              09    A0              12  90 00033            MOVB     #18, 9(P)                             ; 2988
              0A    A0        04    AC  D0 00037            MOVL     VBN, 10(P)                            ; 2989
              0E    A0              52  D0 0003C            MOVL     HIGH_VBN, 14(P)                       ; 2990
                                        04 00040 2$:        RET                                            ; 2992

; Routine Size:  65 bytes,    Routine Base:  CODE + 0731
```

```
;1454    2993  1 %SBTTL 'FREE_PLC_BLOCKS - free placement data'
;1455    2994  1 ROUTINE FREE_PLC_BLOCKS: NOVALUE=
;1456    2995  1
;1457    2996  1 !++
;1458    2997  1 !
;1459    2998  1 ! FUNCTIONAL DESCRIPTION:
;1460    2999  1 !       This routine frees a list of blocks of placement data.
;1461    3000  1 !
;1462    3001  1 ! INPUT PARAMETERS:
;1463    3002  1 !       NONE
;1464    3003  1 !
;1465    3004  1 ! IMPLICIT INPUTS:
;1466    3005  1 !       NONE
;1467    3006  1 !
;1468    3007  1 ! OUTPUT PARAMETERS:
;1469    3008  1 !       NONE
;1470    3009  1 !
;1471    3010  1 ! IMPLICIT OUTPUTS:
;1472    3011  1 !       NONE
;1473    3012  1 !
;1474    3013  1 ! ROUTINE VALUE:
;1475    3014  1 !       NONE
;1476    3015  1 !
;1477    3016  1 ! SIDE EFFECTS:
;1478    3017  1 !       Blocks linked to listhead are freed.
;1479    3018  1 !
;1480    3019  1 !--
;1481    3020  1
;1482    3021  2 BEGIN
;1483    3022  2 LOCAL
;1484    3023  2         P:              REF BBLOCK;     ! Pointer to allocated space
;1485    3024  2
;1486    3025  2
;1487    3026  2 UNTIL REMQUE(.INPUT_PLACEMENT[0], P) DO
;1488    3027  3     BEGIN
;1489    3028  3     FREE_VM(.P[PLC_SIZE], .P);
;1490    3029  2     END;
;1491    3030  2 INPUT_PLACE_LEN = 0;
;1492    3031  2 UNTIL REMQUE(.INPUT_VBN_LIST[0], P) DO
;1493    3032  3     BEGIN
;1494    3033  3     FREE_VM(.P[PLC_SIZE], .P);
;1495    3034  2     END;
;1496    3035  1 END;


                        001C 00000 FREE_PLC_BLOCKS:
                                           .WORD   Save R2,R3,R4                   ; 2994
54 00000000G   00  9E 00002                MOVAB   FREE_VM, R4                     ;
53 00000000'   EF  9E 00009                MOVAB   INPUT_PLACEMENT, R3             ;
52         00  B3  0F 00010 1$:            REMQUE  @INPUT_PLACEMENT, P             ; 3026
               0B  1D 00014                BVS     2$                              ;
               52  DD 00016                PUSHL   P                               ; 3028
7E         09  A2  9A 00018                MOVZBL  9(P), -(SP)                     ;
64             02  FB 0001C                CALLS   #2, FREE_VM                     ;
```

```
                    EF  11 0001F         BRB     1$                              ; 3026
                10  A3  B4 00021 2$:      CLRW    INPUT_PLACE_LEN                 ; 3030
            52  08  B3  0F 00024 3$.      REMQUE  @INPUT_VBN_LIST, P              ; 3031
                    0B  1D 00028         BVS     4$                              ;
                    52  DD 0002A         PUSHL   P                               ; 3033
            7E  09  A2  9A 0002C         MOVZBL  9(P), -(SP)                     ;
            64      02  FB 00030         CALLS   #2, FREE_VM                     ;
                    EF  11 00033         BRB     3$                              ; 3031
                        04 00035 4$:     RET                                     ; 3035

; Routine Size:  54 bytes,    Routine Base:  CODE + 0772
```

```
;  1498          3036  1 %SBTTL 'GET_PLC_DATA - get placement data from file header'
;  1499          3037  1 ROUTINE GET_PLC_DATA(HEADER): NOVALUE=
;  1500          3038  1
;  1501          3039  1 !++
;  1502          3040  1 !
;  1503          3041  1 ! FUNCTIONAL DESCRIPTION:
;  1504          3042  1 !       This routine scans a file header and its extension headers to get
;  1505          3043  1 !       the placement data.
;  1506          3044  1 !
;  1507          3045  1 ! INPUT PARAMETERS:
;  1508          3046  1 !       HEADER          - Pointer to primary header.
;  1509          3047  1 !
;  1510          3048  1 ! IMPLICIT INPUTS:
;  1511          3049  1 !       NONE
;  1512          3050  1 !
;  1513          3051  1 ! OUTPUT PARAMETERS:
;  1514          3052  1 !       NONE
;  1515          3053  1 !
;  1516          3054  1 ! IMPLICIT OUTPUTS:
;  1517          3055  1 !       INPUT_PLACEMENT - List of placement data blocks.
;  1518          3056  1 !       INPUT_PLACE_LEN - Length of placement data as file attribute.
;  1519          3057  1 !       INPUT_VBN_LIST  - For /VOLUME save, list of VBN ranges.
;  1520          3058  1 !       INPUT_FLAGS[INPUT_ON_RVN] - True if any blocks were allocated on the
;  1521          3059  1 !                           selected RVN (even if past INPUT_MAXBLOCK).
;  1522          3060  1 !
;  1523          3061  1 ! ROUTINE VALUE:
;  1524          3062  1 !       NONE
;  1525          3063  1 !
;  1526          3064  1 ! SIDE EFFECTS:
;  1527          3065  1 !       NONE
;  1528          3066  1 !
;  1529          3067  1 !--
;  1530          3068  1
;  1531          3069  2 BEGIN
;  1532          3070  2 LOCAL
;  1533          3071  2         LOCAL_HEADER:   BBLOCK[512],     ! Buffer to read extension headers
;  1534          3072  2         FILE_ID:        BBLOCK[FID$C_LENGTH],   ! Current file ID
;  1535          3073  2         HDR:            REF BBLOCK,      ! Pointer to current file header
;  1536          3074  2         THIS_RVN,                        ! RVN of current header
;  1537          3075  2         THIS_START_VBN,                  ! Starting VBN of current header
;  1538          3076  2         THIS_COUNT,                      ! VBNs mapped by current header
;  1539          3077  2         COUNT;                           ! Count of unplaced blocks
;  1540          3078  2
;  1541          3079  2
;  1542          3080  2 ! Initialize.
;  1543          3081  2 !
;  1544          3082  2 INPUT_PLACEMENT[0] = INPUT_PLACEMENT[1] = INPUT_PLACEMENT[0];
;  1545          3083  2 INPUT_VBN_LIST[0] = INPUT_VBN_LIST[1] = INPUT_VBN_LIST[0];
;  1546          3084  2 INPUT_PLACE_LEN = 0;
;  1547          3085  2 INPUT_FLAGS[INPUT_ON_RVN] = FALSE;
;  1548          3086  2 THIS_RVN = .INPUT_NAM[NAM$B_FID_RVN];
;  1549          3087  2 THIS_START_VBN = 1;
;  1550          3088  2 THIS_COUNT = 0;
;  1551          3089  2 FILE_ID[FID$W_NUM] = .INPUT_NAM[NAM$W_FID_NUM];
;  1552          3090  2 FILE_ID[FID$W_SEQ] = .INPUT_NAM[NAM$W_FID_SEQ];
;  1553          3091  2 FILE_ID[FID$W_RVN] = .INPUT_NAM[NAM$W_FID_RVN];
;  1554          3092  2 HDR = .HEADER;
```

```
;  1555     3093  2 COUNT = 0;
;  1556     3094  2
;  1557     3095  2
;  1558     3096  2 ! Loop for all headers.
;  1559     3097  2 !
;  1560     3098  2 WHILE TRUE DO
;  1561     3099  3     BEGIN
;  1562     3100  3     LOCAL
;  1563     3101  3         MAP_AREA:       REF BBLOCK,     ! Pointer to map area
;  1564     3102  3         MAP_POINTER:    REF BBLOCK,     ! Pointer to map pointer
;  1565     3103  3         MAP_END,                        ! Pointer past map area
;  1566     3104  3         FIB:            BBLOCK[FIB$C_LENGTH],   ! FIB
;  1567     3105  3         FIB_DESC:       VECTOR[2],      ! Descriptor for FIB
;  1568     3106  3         ATR_DESC:       BBLOCK[12],     ! ACP attribute list
;  1569     3107  3         STATUS,                         ! Status variable
;  1570     3108  3         IOSB:           VECTOR[4,WORD], ! I/O status block
;  1571     3109  3         PEND_PLACE,                     ! True if placement pointer pending
;  1572     3110  3         DATA:           BBLOCK[10];     ! Data for placement block
;  1573     3111  3
;  1574     3112  3
;  1575     3113  3     ! Initialize pointers to map area.
;  1576     3114  3     !
;  1577     3115  3     MAP_AREA = .HDR + .HDR[FH2$B_MPOFFSET] * 2;
;  1578     3116  3     IF .HDR[FH2$B_STRUCLEV] EQL 1
;  1579     3117  3     THEN
;  1580     3118  4         BEGIN
;  1581     3119  4         MAP_POINTER = .MAP_AREA + FM1$C_POINTERS;
;  1582     3120  4         MAP_END = .MAP_POINTER + .MAP_AREA[FM1$B_INUSE] * 2;
;  1583     3121  4
;  1584     3122  4
;  1585     3123  4         ! Scan the header to extract placement data.
;  1586     3124  4         !
;  1587     3125  4         WHILE .MAP_POINTER LSSA .MAP_END DO
;  1588     3126  5             BEGIN
;  1589     3127  5             COUNT = .COUNT + .MAP_POINTER[FM1$B_COUNT] + 1;
;  1590     3128  5             THIS_COUNT = .THIS_COUNT + .MAP_POINTER[FM1$B_COUNT] + 1;
;  1591     3129  5             MAP_POINTER = .MAP_POINTER + 4;
;  1592     3130  4             END;
;  1593     3131  4
;  1594     3132  4
;  1595     3133  4         ! Find the next extension header.
;  1596     3134  4         !
;  1597     3135  4         FILE_ID[FID$W_NUM] = .MAP_AREA[FM1$W_EX_FILNUM];
;  1598     3136  4         FILE_ID[FID$W_SEQ] = .MAP_AREA[FM1$W_EX_FILSEQ];
;  1599     3137  4         END
;  1600     3138  3     ELSE
;  1601     3139  4         BEGIN
;  1602     3140  4         MAP_POINTER = .MAP_AREA;
;  1603     3141  4         MAP_END = .MAP_POINTER + .HDR[FH2$B_MAP_INUSE] * 2;
;  1604     3142  4
;  1605     3143  4
;  1606     3144  4         ! Scan the header to extract placement data.
;  1607     3145  4         !
;  1608     3146  4         PEND_PLACE = FALSE;
;  1609     3147  4         WHILE .MAP_POINTER LSSA .MAP_END DO
;  1610     3148  5             BEGIN
;  1611     3149  5             IF .MAP_POINTER[FM2$V_FORMAT] EQL FM2$C_PLACEMENT
```

```
;  1612          3150  5              THEN
;  1613          3151  6                  BEGIN
;  1614          3152  6
;  1615          3153  6                  ! Save placement pointer, and note that one is pending.
;  1616          3154  6                  !
;  1617          3155  6                  DATA[BSA$W_PLC_PTR] = .MAP_POINTER[FM2$W_WORD0];
;  1618          3156  6                  PEND_PLACE = TRUE;
;  1619          3157  6                  MAP_POINTER = .MAP_POINTER + 2;
;  1620          3158  6                  END
;  1621          3159  5              ELSE
;  1622          3160  6                  BEGIN
;  1623          3161  6
;  1624          3162  6                  ! Extract count and LBN of retrieval pointer.
;  1625          3163  6                  !
;  1626          3164  6                  CASE .MAP_POINTER[FM2$V_FORMAT] FROM FM2$C_FORMAT1 TO FM2$C_FORMAT3 OF
;  1627          3165  6                      SET
;  1628          3166  6
;  1629          3167  6                      [FM2$C_FORMAT1]:
;  1630          3168  7                          BEGIN
;  1631          3169  7                          DATA[BSA$L_PLC_COUNT] = .MAP_POINTER[FM2$B_COUNT1];
;  1632          3170  7                          DATA[BSA$W_PLC_HILBN] = .MAP_POINTER[FM2$V_HIGHLBN];
;  1633          3171  7                          DATA[BSA$W_PLC_LOLBN] = .MAP_POINTER[FM2$W_LOWLBN];
;  1634          3172  7                          MAP_POINTER = .MAP_POINTER + 4;
;  1635          3173  6                          END;
;  1636          3174  6
;  1637          3175  6                      [FM2$C_FORMAT2]:
;  1638          3176  7                          BEGIN
;  1639          3177  7                          DATA[BSA$L_PLC_COUNT] = .MAP_POINTER[FM2$V_COUNT2];
;  1640          3178  7                          DATA[BSA$L_PLC_LBN] = .MAP_POINTER[FM2$L_LBN2];
;  1641          3179  7                          MAP_POINTER = .MAP_POINTER + 6;
;  1642          3180  6                          END;
;  1643          3181  6
;  1644          3182  6                      [FM2$C_FORMAT3]:
;  1645          3183  7                          BEGIN
;  1646          3184  7                          DATA[BSA$L_PLC_COUNT] = ROT(..MAP_POINTER, 16) AND (1^30-1);
;  1647          3185  7                          DATA[BSA$L_PLC_LBN] = .MAP_POINTER[FM2$L_LBN3];
;  1648          3186  7                          MAP_POINTER = .MAP_POINTER + 8;
;  1649          3187  6                          END;
;  1650          3188  6
;  1651          3189  6                      TES;
;  1652          3190  6              DATA[BSA$L_PLC_COUNT] = .DATA[BSA$L_PLC_COUNT] + 1;
;  1653          3191  6              THIS_COUNT = .THIS_COUNT + .DATA[BSA$L_PLC_COUNT];
;  1654          3192  6
;  1655          3193  6
;  1656          3194  6              IF .PEND_PLACE
;  1657          3195  6              THEN
;  1658          3196  7                  BEGIN
;  1659          3197  7                  IF .COUNT NEQ 0
;  1660          3198  7                  THEN
;  1661          3199  8                      BEGIN
;  1662          3200  8                      ALLOC_PLC_BLOCK(BSA$K_PLC_COUNT, BSA$S_PLC_COUNT, COUNT);
;  1663          3201  8                      COUNT = 0;
;  1664          3202  7                      END;
;  1665          3203  7                  IF .DATA[FM2$V_LBN]
;  1666          3204  7                  THEN
;  1667          3205  7                      ALLOC_PLC_BLOCK(BSA$K_PLC_PLLBN, BSA$S_PLC_PLLBN, DATA)
;  1668          3206  7                  ELSE
```

```
;  1669          3207  7                        ALLOC_PLC_BLOCK(BSA$K_PLC_PLACE, BSA$S_PLC_PLACE, DATA);
;  1670          3208  7                    PEND_PLACE = FALSE;
;  1671          3209  7                    END
;  1672          3210  6                ELSE
;  1673          3211  6                    COUNT = .COUNT + .DATA[BSA$L_PLC_COUNT];
;  1674          3212  5                END;
;  1675          3213  4            END;
;  1676          3214  4
;  1677          3215  4
;  1678          3216  4        ! Find the next extension header.
;  1679          3217  4        !
;  1680          3218  4        FILE_ID[FID$W_NUM] = .HDR[FH2$W_EX_FIDNUM];
;  1681          3219  4        FILE_ID[FID$B_NMX] = .HDR[FH2$B_EX_FIDNMX];
;  1682          3220  4        FILE_ID[FID$W_SEQ] = .HDR[FH2$W_EX_FIDSEQ];
;  1683          3221  4        IF .HDR[FH2$B_EX_FIDRVN] NEQ 0
;  1684          3222  4            THEN FILE_ID[FID$B_RVN] = .HDR[FH2$B_EX_FIDRVN];
;  1685          3223  3        END;
;  1686          3224  3
;  1687          3225  3
;  1688          3226  3    ! If there is no extension header, we are done.
;  1689          3227  3    !
;  1690          3228  3    IF .FILE_ID[FID$W_NUM] EQL 0
;  1691          3229  3    THEN
;  1692          3230  3        EXITLOOP;
;  1693          3231  3
;  1694          3232  3
;  1695          3233  3    ! Put out a placement data block for the extension file ID.
;  1696          3234  3    !
;  1697          3235  3    IF .COUNT NEQ 0
;  1698          3236  3    THEN
;  1699          3237  4        BEGIN
;  1700          3238  4        ALLOC_PLC_BLOCK(BSA$K_PLC_COUNT, BSA$S_PLC_COUNT, COUNT);
;  1701          3239  4        COUNT = 0;
;  1702          3240  3        END;
;  1703          3241  3    ALLOC_PLC_BLOCK(BSA$K_PLC_FID, BSA$S_PLC_FID, FILE_ID);
;  1704          3242  3
;  1705          3243  3
;  1706          3244  3    ! If the RVN changes, and if required, put out a VBN block for the
;  1707          3245  3    ! sequence of blocks.
;  1708          3246  3    !
;  1709          3247  3    IF .THIS_RVN NEQ .FILE_ID[FID$B_RVN]
;  1710          3248  3    THEN
;  1711          3249  4        BEGIN
;  1712          3250  4        IF .QUAL[QUAL_VOLU] AND .QUAL[QUAL_VOLU_VALUE] EQL .THIS_RVN
;  1713          3251  4        THEN
;  1714          3252  4            ALLOC_VBN_BLOCK(.THIS_START_VBN, .THIS_COUNT);
;  1715          3253  4        THIS_START_VBN = .THIS_START_VBN + .THIS_COUNT;
;  1716          3254  4        THIS_COUNT = 0;
;  1717          3255  4        THIS_RVN = .FILE_ID[FID$B_RVN];
;  1718          3256  3        END;
;  1719          3257  3
;  1720          3258  3
;  1721          3259  3    ! Read the extension header.                                                                    0
;  1722          3260  3    !
;  1723          3261  3    CH$FILL (0, FIB$C_LENGTH, FIB);
;  1724          3262  3    FIB[FIB$L_ACCTL] = 0;
;  1725          3263  3    FIB[FIB$W_FID_NUM] = .FILE_ID[FID$W_NUM];
```

```
;  1726          3264  3     FIB[FIB$W_FID_SEQ] = .FILE_ID[FID$W_SEQ];
;  1727          3265  3     FIB[FIB$W_FID_RVN] = .FILE_ID[FID$W_RVN];
;  1728          3266  3     FIB_DESC[0] = FIB$C_LENGTH;
;  1729          3267  3     FIB_DESC[1] = FIB;
;  1730          3268  3     ATR_DESC[0,0,16,0] = ATR$S_HEADER;
;  1731          3269  3     ATR_DESC[2,0,16,0] = ATR$C_HEADER;
;  1732          3270  3     ATR_DESC[4,0,32,0] = LOCAL_HEADER;
;  1733          3271  3     ATR_DESC[8,0,32,0] = 0;
;  1734        P 3272  3     STATUS = $QIOW(
;  1735        P 3273  3         FUNC=IO$_ACCESS,
;  1736        P 3274  3         CHAN=.INPUT_CHAN,
;  1737        P 3275  3         IOSB=IOSB,
;  1738        P 3276  3         P1=FIB_DESC,
;  1739          3277  3         P5=ATR_DESC);
;  1740          3278  3     IF .STATUS THEN STATUS = .IOSB[0];
;  1741          3279  3     IF NOT .STATUS
;  1742          3280  3     THEN
;  1743          3281  4         BEGIN
;  1744          3282  4         FILE_ERROR(BACKUP$_READATTR, .INPUT_FAB, .STATUS);
;  1745          3283  4         FREE_PLC_BLOCKS();
;  1746          3284  4         RETURN;
;  1747          3285  3         END;
;  1748          3286  3     HDR = LOCAL_HEADER;
;  1749          3287  2     END;
;  1750          3288  2
;  1751          3289  2
;  1752          3290  2 ! If required, put out a VBN block for the last sequence of blocks.
;  1753          3291  2 !
;  1754          3292  2 IF .QUAL[QUAL_VOLU] AND .QUAL[QUAL_VOLU_VALUE] EQL .THIS_RVN
;  1755          3293  2 THEN
;  1756          3294  2     ALLOC_VBN_BLOCK(.THIS_START_VBN, .THIS_COUNT);
;  1757          3295  2
;  1758          3296  2
;  1759          3297  2 ! If required, put out a placement data block for unplaced blocks.
;  1760          3298  2 !
;  1761          3299  2 IF .COUNT NEQ 0 AND .INPUT_PLACEMENT[0] NEQ INPUT_PLACEMENT[0]
;  1762          3300  2 THEN
;  1763          3301  2     ALLOC_PLC_BLOCK(BSA$K_PLC_COUNT, BSA$S_PLC_COUNT, COUNT);
;  1764          3302  1 END;


                                  0FFC 00000 GET_PLC_DATA:
                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 3037
                5B      FF4C    CF   9E 00002           MOVAB   ALLOC_PLC_BLOCK, R11                  ;
                5A 00000000'    EF   9E 00007           MOVAB   INPUT_PLACEMENT, R10                  ;
                5E      FD90    CE   9E 0000E           MOVAB   -624(SP), SP                          ;
                50              6A   9E 00013           MOVAB   INPUT_PLACEMENT, R0                   ; 3082
          04    AA              50   D0 00016           MOVL    R0, INPUT_PLACEMENT+4                 ;
                6A              50   D0 0001A           MOVL    R0, INPUT_PLACEMENT                   ;
                50        08    AA   9E 0001D           MOVAB   INPUT_VBN_LIST, R0                    ; 3083
          0C    AA              50   D0 00021           MOVL    R0, INPUT_VBN_LIST+4                  ;
          08    AA              50   D0 00025           MOVL    R0, INPUT_VBN_LIST                    ;
                          10    AA   B4 00029           CLRW    INPUT_PLACE_LEN                       ; 3084
          80    AA              08   8A 0002C           BICB2   #8, INPUT_FLAGS                       ; 3085
```

```
                                50       88    AA   D0 00030          MOVL    INPUT_NAM, R0                                   ; 3086
                                58       28    A0   9A 00034          MOVZBL  40(R0), THIS_RVN                                ;
                                59             01   D0 00038          MOVL    #1, THIS_START_VBN                              ; 3087
                                               57   D4 0003B          CLRL    THIS_COUNT                                      ; 3088
                            68  AE       24    A0   D0 0003D          MOVL    36(R0), FILE_ID                                 ; 3089
                            6C  AE       28    A0   B0 00042          MOVW    40(R0), FILE_ID+4                               ; 3091
                                56       04    AC   D0 00047          MOVL    HEADER, HDR                                     ; 3092
                                               7E   D4 0004B          CLRL    COUNT                                           ; 3093
                                50       01    A6   9A 0004D 1$:      MOVZBL  1(HDR), R0                                      ; 3115
                                50           6640   3E 00051          MOVAW   (HDR)[R0], MAP_AREA                             ;
                                01       07    A6   91 00055          CMPB    7(HDR), #1                                      ; 3116
                                               2F   12 00059          BNEQ    4$                                              ;
                                52       0A    A0   9E 0005B          MOVAB   10(R0), MAP_POINTER                             ; 3119
                                51       08    A0   9A 0005F          MOVZBL  8(MAP_AREA), R1                                 ; 3120
                                54           6241   3E 00063          MOVAW   (MAP_POINTER)[R1], MAP_END                      ;
                                54             52   D1 00067 2$:      CMPL    MAP_POINTER, MAP_END                            ; 3125
                                               16   1E 0006A          BGEQU   3$                                              ;
                                51       01    A2   9A 0006C          MOVZBL  1(MAP_POINTER), R1                              ; 3127
                        53      6E             51   C1 00070          ADDL3   R1, COUNT, R3                                   ;
                                6E       01    A3   9E 00074          MOVAB   1(R3), COUNT                                    ;
                                57       01 A147   9E 00078          MOVAB   1(R1)[THIS_COUNT], THIS_COUNT                   ; 3128
                                52             04   C0 0007D          ADDL2   #4, MAP_POINTER                                 ; 3129
                                               E5   11 00080          BRB     2$                                              ; 3125
                            6C  AE       02    A0   D0 00082 3$:      MOVL    2(MAP_AREA), FILE_ID                            ; 3135
                                           00B3   31 00087          BRW     20$                                             ; 3116
                                52             50   D0 0008A 4$:      MOVL    MAP_AREA, MAP_POINTER                           ; 3140
                                50       3A    A6   9A 0008D          MOVZBL  58(ADR), R0                                     ; 3141
                                54           6240   3E 00091          MOVAW   (MAP_POINTER)[R0], MAP_END                      ;
                                               53   D4 00095 5$:      CLRL    PEND_PLACE                                      ; 3146
                                54             52   D1 00097 6$:      CMPL    MAP_POINTER, MAP_END                            ; 3147
                                               03   1F 0009A          BLSSU   7$                                              ;
                                           008A   31 0009C          BRW     19$                                             ;
                            C0  8F       01    A2   93 0009F 7$:      BITB    1(MAP_POINTER), #192                            ; 3149
                                               09   12 000A4          BNEQ    9$                                              ;
                            04  AE             82   B0 000A6          MOVW    (MAP_POINTER)+, DATA                            ; 3155
                                53             01   D0 000AA          MOVL    #1, PEND_PLACE                                  ; 3156
                                               E8   11 000AD 8$:      BRB     6$                                              ; 3149
                    51          62             0E   EF 000AF 9$:      EXTZV   #14, #2, (MAP_POINTER), R1                      ; 3164
                                02             51   CF 000B4          CASEL   R1, #1, #2                                      ;
                              0029           0006       000B8 10$:     .WORD   11$-10$,-                                       ;
                                                                                12$-10$,-                                       ;
                                                                                13$-10$                                         ;
                            06  AE             82   9A 000BE 11$:     MOVZBL  (MAP_POINTER)+, DATA+2                          ; 3169
                    50          82             00   EF 000C2          EXTZV   #0, #6, (MAP_POINTER)+, R0                      ; 3170
                            0C  AE             50   B0 000C7          MOVW    R0, DATA+8                                      ;
                            0A  AE             82   B0 000CB          MOVW    (MAP_POINTER)+, DATA+6                          ; 3171
                                               1E   11 000CF          BRB     14$                                             ; 3164
                06  AE          82             00   EF 000D1 12$:     EXTZV   #0, #14, (MAP_POINTER)+, DATA+2                 ; 3177
                            0A  AE       01    A2   D0 000D7          MOVL    1(MAP_POINTER), DATA+6                          ; 3178
                                52             05   C0 000DC          ADDL2   #5, MAP_POINTER                                 ; 3179
                                               0E   11 000DF          BRB     14$                                             ; 3164
                                50             82   10   9C 000E1 13$:     ROTL    #16, (MAP_POINTER)+, R0                 ; 3184
                06  AE          50             1E   00   EF 000E5          EXTZV   #0, #30, R0, DATA+2                             ;
                            0A  AE             82   D0 000EB          MOVL    (MAP_POINTER)+, DATA+6                          ; 3185
                                           06  AE   D6 000EF 14$:     INCL    DATA+2                                          ; 3190
                                57         06  AE   C0 000F2          ADDL2   DATA+2, THIS_COUNT                              ; 3191
                                2A             53   E9 000F6          BLBC    PEND_PLACE, T8$                                 ; 3194
```

```
                                                 6E  D5 000F9          TSTL    COUNT                                          ; 3197
                                                 0B  13 000FB          BEQL    15$                                            ;
                                                 5E  DD 000FD          PUSHL   SP                                             ; 3200
                                                 04  DD 000FF          PUSHL   #4                                             ;
                                                 02  DD 00101          PUSHL   #2                                             ;
                                     6B          03  FB 00103          CALLS   #3, ALLOC_PLC_BLOCK                            ;
                                                 6E  D4 00106          CLRL    COUNT                                          ; 3201
                                09       05  AE     04  E1 00108 15$:   BBC     #4, DATA+1, 16$                                ; 3203
                                             04  AE  9F 0010D          PUSHAB  DATA                                           ; 3205
                                                 0A  DD 00110          PUSHL   #10                                            ;
                                                 04  DD 00112          PUSHL   #4                                             ;
                                                 07  11 00114          BRB     17$                                            ;
                                             04  AE  9F 00116 16$:   PUSHAB  DATA                                           ; 3207
                                                 06  DD 00119          PUSHL   #6                                             ;
                                                 03  DD 0011B          PUSHL   #3                                             ;
                                     6B          03  FB 0011D 17$:   CALLS   #3, ALLOC_PLC_BLOCK                            ;
                                               FF72  31 00120          BRW     5$                                             ; 3208
                                     6E      06  AE  C0 00123 18$:   ADDL2   DATA+2, COUNT                                  ; 3211
                                                 84  11 00127          BRB     8$                                             ; 3147
                                 71  AE      13  A6  90 00129 19$:   MOVB    19(HDR), FILE_ID+5                             ; 3219
                                 6C  AE      0E  A6  D0 0012E          MOVL    14(HDR), FILE_ID                               ; 3218
                                             12  A6  95 00133          TSTB    18(HDR)                                        ; 3221
                                                 05  13 00136          BEQL    20$                                            ;
                                 70  AE      12  A6  90 00138          MOVB    18(HDR), FILE_ID+4                             ; 3222
                                             6C  AE  B5 0013D 20$:   TSTW    FILE_ID                                        ; 3228
                                                 03  12 00140          BNEQ    21$                                            ;
                                               00B8  31 00142          BRW     27$                                            ;
                                                 6E  D5 00145 21$:   TSTL    COUNT                                          ; 3235
                                                 0B  13 00147          BEQL    22$                                            ;
                                                 5E  DD 00149          PUSHL   SP                                             ; 3238
                                                 04  DD 0014B          PUSHL   #4                                             ;
                                                 02  DD 0014D          PUSHL   #2                                             ;
                                     6B          03  FB 0014F          CALLS   #3, ALLOC_PLC_BLOCK                            ;
                                                 6E  D4 00152          CLRL    COUNT                                          ; 3239
                                             6C  AE  9F 00154 22$:   PUSHAB  FILE_ID                                        ; 3241
                                                 06  DD 00157          PUSHL   #6                                             ;
                                                 01  DD 00159          PUSHL   #1                                             ;
                                     6B          03  FB 0015B          CALLS   #3, ALLOC_PLC_BLOCK                            ;
        58       70  AE          08              00  ED 0015E          CMPZV   #0, #8, FILE_ID+4, THIS_RVN                    ; 3247
                                                 1F  13 00164          BEQL    24$                                            ;
                                     11      FEC6  CA  E9 00166          BLBC    QUAL+14, 23$                                   ; 3250
        58     FF07  CA              08              00  ED 0016B          CMPZV   #0, #8, QUAL+79, THIS_RVN                      ;
                                                 08  12 00172          BNEQ    23$                                            ;
                                                 57  DD 00174          PUSHL   THIS_COUNT                                     ; 3252
                                                 59  DD 00176          PUSHL   THIS_START_VBN                                 ;
                                 37  AB          02  FB 00178          CALLS   #2, ALLOC_VBN_BLOCK                            ;
                                     59          57  C0 0017C 23$:   ADDL2   THIS_COUNT, THIS_START_VBN                    ; 3253
                                                 57  D4 0017F          CLRL    THIS_COUNT                                     ; 3254
                                     58      70  AE  9A 00181          MOVZBL  FILE_ID+4, THIS_RVN                            ; 3255
  0040  8F          00             6E              00  2C 00185 24$:   MOVC5   #0, (SP), #0, #64, FIB                       ; 3261
                                             2C  AE      0018C                                                              ;
                                             2C  AE  D4 0018E          CLRL    FIB                                            ; 3262
                                 30  AE      6C  AE  D0 00191          MOVL    FILE_ID, FIB+4                                 ; 3263
                                 34  AE      70  AE  B0 00196          MOVW    FILE_ID+4, FIB+8                               ; 3265
                                 24  AE      40  8F  9A 0019B          MOVZBL  #64, FIB_DESC                                  ; 3266
                                 28  AE      2C  AE  9E 001A0          MOVAB   FIB, FIB_DESC+4                                ; 3267
                                 18  AE  000A0200  8F  D0 001A5          MOVL    #655872, ATR_DESC                              ; 3268
```

```
                          1C    AE        74    AE    9E 001AD          MOVAB   LOCAL_HEADER, ATR_DESC+4                ; 3270
                                          20    AE    D4 001B2          CLRL    ATR_DESC+8                              ; 3271
                                                7E    D4 001B5          CLRL    -(SP)                                   ; 3277
                                          1C    AE    9F 001B7          PUSHAB  ATR_DESC                                ;
                                                7E    7C 001BA          CLRQ    -(SP)                                   ;
                                                7E    D4 001BC          CLRL    -(SP)                                   ;
                                          38    AE    9F 001BE          PUSHAB  FIB_DESC                                ;
                                                7E    7C 001C1          CLRQ    -(SP)                                   ;
                                          30    AE    9F 001C3          PUSHAB  IOSB                                    ;
                                                32    DD 001C6          PUSHL   #50                                     ;
                                        FF7C    CA    DD 001C8          PUSHL   INPUT_CHAN                              ;
                                                7E    D4 001CC          CLRL    -(SP)                                   ;
                  00000000G             00          0C    FB 001CE          CALLS   #12, SYS$QIOW                       ;
                                        07          50    E9 001D5          BLBC    STATUS, 25$                             ; 3278
                                        50          10    AE    3C 001D8    MOVZWL  IOSB, STATUS                            ;
                                        17                50    E8 001DC    BLBS    STATUS, 26$                             ; 3279
                                                          50    DD 001DF 25$: PUSHL   STATUS                              ; 3282
                                                    84    AA    DD 001E1    PUSHL   INPUT_FAB                           ;
                                           00000000G      8F    DD 001E4    PUSHL   #BACKUP$_READATTR                   ;
                  00000000G             00                03    FB 001EA    CALLS   #3, FILE_ERROR                      ;
                          78    AB                        00    FB 001F1    CALLS   #0, FREE_PLC_BLOCKS                 ; 3283
                                                          04    001F5       RET                                         ; 3281
                                        56        74    AE    9E 001F6 26$: MOVAB   LOCAL_HEADER, HDR                   ; 3286
                                                FE50      31    001FA       BRW     1$                                  ; 3098
                                        11      FEC6      CA    E9 001FD 27$: BLBC    QUAL+14, 28$                        ; 3292
          58    FF07    CA                    08              00    ED 00202    CMPZV   #0, #8, QUAL+79, THIS_RVN            ;
                                                              08    12 00209    BNEQ    28$                                 ;
                                                              57    DD 0020B    PUSHL   THIS_COUNT                          ; 3294
                                                              59    DD 0020D    PUSHL   THIS_START_VBN                      ;
                                                    37    AB    02    FB 0020F    CALLS   #2, ALLOC_VBN_BLOCK             ;
                                                              6E    D5 00213 28$: TSTL    COUNT                           ; 3299
                                                              11    13 00215    BEQL    29$                             ;
                                                        50    6A    9E 00217    MOVAB   INPUT_PLACEMENT, R0             ;
                                                        50    6A    D1 0021A    CMPL    INPUT_PLACEMENT, R0             ;
                                                              09    13 0021D    BEQL    29$                             ;
                                                              5E    DD 0021F    PUSHL   SP                              ; 3301
                                                              04    DD 00221    PUSHL   #4                              ;
                                                              02    DD 00223    PUSHL   #2                              ;
                                                        6B    03    FB 00225    CALLS   #3, ALLOC_PLC_BLOCK             ;
                                                                    04 00228 29$: RET                                   ; 3302

; Routine Size:  553 bytes,    Routine Base:  CODE + 07A8
```

```
;  1766          3303  1 %SBTTL 'DUMP_PLC_DATA - format placement data into attribute record'
;  1767          3304  1 ROUTINE DUMP_PLC_DATA(ATT_AREA): NOVALUE=
;  1768          3305  1
;  1769          3306  1 !++
;  1770          3307  1 !
;  1771          3308  1 ! FUNCTIONAL DESCRIPTION:
;  1772          3309  1 !       This routine puts the placement data into the file attribute record.
;  1773          3310  1 !
;  1774          3311  1 ! INPUT PARAMETERS:
;  1775          3312  1 !       ATT_AREA        - Pointer to area in file attribute record.
;  1776          3313  1 !
;  1777          3314  1 ! IMPLICIT INPUTS:
;  1778          3315  1 !       NONE
;  1779          3316  1 !
;  1780          3317  1 ! OUTPUT PARAMETERS:
;  1781          3318  1 !       NONE
;  1782          3319  1 !
;  1783          3320  1 ! IMPLICIT OUTPUTS:
;  1784          3321  1 !       NONE
;  1785          3322  1 !
;  1786          3323  1 ! ROUTINE VALUE:
;  1787          3324  1 !       NONE
;  1788          3325  1 !
;  1789          3326  1 ! SIDE EFFECTS:
;  1790          3327  1 !       NONE
;  1791          3328  1 !
;  1792          3329  1 !--
;  1793          3330  1
;  1794          3331  2 BEGIN
;  1795          3332  2 LOCAL
;  1796          3333  2         P:              REF BBLOCK,     ! Pointer to placement data block
;  1797          3334  2         A;                              ! Pointer to attribute area
;  1798          3335  2
;  1799          3336  2
;  1800          3337  2 A = .ATT_AREA;
;  1801          3338  2 P = .INPUT_PLACEMENT[0];
;  1802          3339  2 UNTIL .P EQL INPUT_PLACEMENT[0] DO
;  1803          3340  3     BEGIN
;  1804          3341  3     (.A)<0,8> = .P[PLC_TYPE];
;  1805          3342  3     A = .A + 1;
;  1806          3343  3     A = CH$MOVE(.P[PLC_SIZE] - PLC_S_HDR, P[PLC_DATA], .A);
;  1807          3344  3     P = .P[PLC_FLINK];
;  1808          3345  2     END;
;  1809          3346  1 END;

                                00FC 00000 DUMP_PLC_DATA:
                                           .WORD   Save R2,R3,R4,R5,R6,R7                  ; 3304
                57 00000000'    EF  9E 00002       MOVAB   INPUT_PLACEMENT, R7             ;
                53        04    AC  D0 00009       MOVL    ATT_AREA, A                     ; 3337
                56              67  D0 0000D       MOVL    INPUT_PLACEMENT, P              ; 3338
                50              67  9E 00010 1$:   MOVAB   INPUT_PLACEMENT, R0             ; 3339
                50              56  D1 00013       CMPL    P, R0                           ;
                                15  13 00016       BEQL    2$                              ;
```

```
                                83      08    A6  90 00018          MOVB    8(P), (A)+                              ; 3341
                                50      09    A6  9A 0001C          MOVZBL  9(P), R0                                ; 3343
                                50            0A  C2 00020          SUBL2   #10, R0                                 ;
                        63      0A    A6        50  28 00023          MOVC3   R0, 10(P), (A)                          ;
                                56            66  D0 00028          MOVL    (P), P                                  ; 3344
                                              E3  11 0002B          BRB     1$                                      ; 3339
                                                  04 0002D 2$:      RET                                             ; 3346

; Routine Size:  46 bytes,    Routine Base:  CODE + 09D1
```

```
;1811    3347  1 %SBTTL 'FILE_ATTRIBUTES - format file attribute record'
;1812    3348  1 ROUTINE FILE_ATTRIBUTES(INPUT_HDR): NOVALUE=
;1813    3349  1
;1814    3350  1 !++
;1815    3351  1 !
;1816    3352  1 ! FUNCTIONAL DESCRIPTION:
;1817    3353  1 !       This routine generates a file attribute record.
;1818    3354  1 !
;1819    3355  1 ! INPUT PARAMETERS:
;1820    3356  1 !       INPUT_HDR       - Pointer to file header.
;1821    3357  1 !
;1822    3358  1 ! IMPLICIT INPUTS:
;1823    3359  1 !       NONE
;1824    3360  1 !
;1825    3361  1 ! OUTPUT PARAMETERS:
;1826    3362  1 !       NONE
;1827    3363  1 !
;1828    3364  1 ! IMPLICIT OUTPUTS:
;1829    3365  1 !       NONE
;1830    3366  1 !
;1831    3367  1 ! ROUTINE VALUE:
;1832    3368  1 !       NONE
;1833    3369  1 !
;1834    3370  1 ! SIDE EFFECTS:
;1835    3371  1 !       NONE
;1836    3372  1 !
;1837    3373  1 !--
;1838    3374  1
;1839    3375  2 BEGIN
;1840    3376  2 MAP
;1841    3377  2         INPUT_HDR:      REF BBLOCK;     ! Pointer to file header
;1842    3378  2 LOCAL
;1843    3379  2         L1,                             ! Unrounded length
;1844    3380  2         L2,                             ! Rounded length
;1845    3381  2         P:              REF BBLOCK,     ! Pointer to attribute record
;1846    3382  2         INPUT_IDENT:    REF BBLOCK,     ! Pointer to ident area
;1847    3383  2         IDLEN,                          ! Length of ident area
;1848    3384  2         BOOTVBN,                        ! Boot VBN
;1849    3385  2         PLACEMENT,                      ! Generating placement attribute
;1850    3386  2         SAVE_ACL_PRI,                   ! Save ACL in primary attr record
;1851    3387  2         SAVE_ACL_EXT,                   ! Save ACL in extension attr record
;1852    3388  2         ATR_DESC        : BBLOCK [20],  ! ACP attribute list
;1853    3389  2         FIB_DESC        : $BBLOCK [DSC$C_S_BLN],        ! FIB descriptor
;1854    3390  2         FIB_ACL         : $BBLOCK [FIB$C_LENGTH],       ! FIB for ACL use
;1855    3391  2         ACL_SEGMENT_SIZE,               ! Size of current ACL segment
;1856    3392  2         STATUS,                         ! Status return
;1857    3393  2         IOSB            : VECTOR [4,WORD];      ! I/O status block
;1858    3394  2 LITERAL
;1859    3395  2     FIXED_SIZE =
;1860    3396  2         BRH$C_LENGTH +          ! record header
;1861    3397  2         2 +                     ! structure level
;1862    3398  2         8*4 +                   ! 8 attribute size/type longwords
;1863    3399  2                                 !  1. file name (variable)
;1864    3400  2         BSA$S_STRUCLEV +        !  2. file structure level
;1865    3401  2         BSA$S_FID +             !  3. file ID
;1866    3402  2         BSA$S_FILESIZE +        !  4. file size in blocks
;1867    3403  2         BSA$S_UIC +             !  5. file owner UIC
```

```
            BSA$S_FPRO +                ! 6. file protection mask
            BSA$S_UCHAR +               ! 7. file characteristics
            BSA$S_RECATTR;              ! 8. record attributes area


    ! Determine the address and length of the ident area.
    !
    INPUT_IDENT = .INPUT_HDR + .INPUT_HDR[FH2$B_IDOFFSET] * 2;
    IDLEN = (.INPUT_HDR[FH2$B_MPOFFSET] - .INPUT_HDR[FH2$B_IDOFFSET]) * 2;


    ! Determine the length of the attribute record.
    !
    L1 = FIXED_SIZE + .INPUT_NAMEDESC[DSC$W_LENGTH];
    IF .INPUT_HDR[FH2$B_STRUCLEV] EQL 2
    THEN
        BEGIN
        L1 = .L1 + 4*5 + BSA$S_BACKLINK + BSA$S_RPRO + BSA$S_ACLEVEL +
             BSA$S_VERLIMIT + BSA$S_JNL_FLAGS;
        IF .INPUT_HDR[FH2$B_IDOFFSET] GEQU ($BYTEOFFSET(FH2$L_HIGHWATER)+4)/2
            THEN L1 = .L1 + 4 + BSA$S_HIGHWATER;
        IF .IDLEN GEQU $BYTEOFFSET(FI2$W_REVISION)+2
            THEN L1 = .L1 + 4 + BSA$S_REVISION;
        IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_CREDATE)+8
            THEN L1 = .L1 + 4 + BSA$S_CREDATE;
        IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_REVDATE)+8
            THEN L1 = .L1 + 4 + BSA$S_REVDATE;
        IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_EXPDATE)+8
            THEN L1 = .L1 + 4 + BSA$S_EXPDATE;
        IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_BAKDATE)+8
            THEN L1 = .L1 + 4 + BSA$S_BAKDATE;
        END
    ELSE
        BEGIN
        L1 = .L1 + 4*4 + BSA$S_REVISION + BSA$S_CREDATE + BSA$S_REVDATE + BSA$S_EXPDATE;


        ! Determine if the file is a directory.  In ODS-1, this is defined as a
        ! file ".DIR;1" with fixed length 16 byte records and no record attributes.
        !
        IF
            CH$FIND_SUB(
                .INPUT_NAMEDESC[DSC$W_LENGTH], .INPUT_NAMEDESC[DSC$A_POINTER],
                6, UPLIT BYTE ('.DIR;1')) NEQ 0 AND
            .INPUT_RECATTR[FAT$B_RTYPE] EQL FAT$C_FIXED AND
            .INPUT_RECATTR[FAT$B_RATTRIB] EQL 0 AND
            .INPUT_RECATTR[FAT$W_RSIZE] EQL NMB$C_DIRENTRY
        THEN
            INPUT_FILECHAR[FCH$V_DIRECTORY] = TRUE;
        END;


    BOOTVBN = 0;
    IF .QUAL[QUAL_IMAG]
    THEN
        IF
            .INPUT_STATBLK[SBK$L_STLBN] NEQ 0 AND
```

```
;  1925          3461  2          .FAST_BOOT_LBN[.INPUT_NAM[NAM$B_FID_RVN]-1] NEQ -1 AND
;  1926          3462  2          .INPUT_STATBLK[SBK$L_STLBN] LEQU .FAST_BOOT_LBN[.INPUT_NAM[NAM$B_FID_RVN]-1] AND
;  1927          3463  2          .INPUT_STATBLK[SBK$L_STLBN] + .INPUT_STATBLK[SBK$L_FILESIZE] GTRU .FAST_BOOT_LBN[.INPUT_NAM[NAM$B_FI
;  1928          3464  2      THEN
;  1929          3465  3          BEGIN
;  1930          3466  3          BOOTVBN = .FAST_BOOT_LBN[.INPUT_NAM[NAM$B_FID_RVN]-1] - .INPUT_STATBLK[SBK$L_STLBN] + 1;
;  1931          3467  3          L1 = .L1 + 4 + BSA$S_BOOTVBN;
;  1932          3468  2          END;
;  1933          3469  2
;  1934          3470  2
;  1935          3471  2  PLACEMENT = FALSE;
;  1936          3472  2  IF
;  1937          3473  2      .QUAL[QUAL_IMAG] AND
;  1938          3474  2      .INPUT_PLACE_LEN NEQ 0 AND
;  1939          3475  2      .L1 + .INPUT_PLACE_LEN + 4 LEQU MAX_RECORD
;  1940          3476  2  THEN
;  1941          3477  3      BEGIN
;  1942          3478  3      L1 = .L1 + .INPUT_PLACE_LEN + 4;
;  1943          3479  3      PLACEMENT = TRUE;
;  1944          3480  2      END;
;  1945          3481  2
;  1946          3482  2
;  1947          3483  2  IF .INPUT_NAM[NAM$W_DID_NUM] NEQ 0
;  1948          3484  2  THEN
;  1949          3485  2      L1 = .L1 + 3*4 + BSA$S_DIR_UIC + BSA$S_DIR_FPRO + BSA$S_DIR_VERLIM;
;  1950          3486  2
;  1951          3487  2
;  1952          3488  2  IF .DIR_STATUS NEQ 0
;  1953          3489  2  THEN
;  1954          3490  2      L1 = .L1 + 4 + BSA$S_DIR_STATUS;
;  1955          3491  2
;  1956          3492  2
;  1957          3493  2  SAVE_ACL_PRI = SAVE_ACL_EXT = FALSE;
;  1958          3494  2  IF .ACL_LENGTH GTR 0
;  1959          3495  2  AND NOT .QUAL[QUAL_INTE]                          ! Ignore ACL's if /INTERCHANGE
;  1960          3496  2  AND .INPUT_HDR[FH2$B_STRUCLEV] EQL 2              ! Only for ODS-2 files
;  1961          3497  2  THEN
;  1962          3498  3      BEGIN
;  1963          3499  3      IF .L1 + 4 + .ACL_LENGTH LEQU MAX_RECORD
;  1964          3500  3      THEN
;  1965          3501  4          BEGIN
;  1966          3502  4          L1 = .L1 + 4 + .ACL_LENGTH;
;  1967          3503  4          SAVE_ACL_PRI = TRUE;
;  1968          3504  4          END
;  1969          3505  3      ELSE SAVE_ACL_EXT = TRUE;
;  1970          3506  2      END;
;  1971          3507  2
;  1972          3508  2
;  1973          3509  2  ! Allocate the attribute record and initialize the record header.
;  1974          3510  2  !
;  1975          3511  2  L2 = (.L1 + 15) AND NOT 15;
;  1976          3512  2  P = MAKE_SPACE(.L2);
;  1977          3513  2  P[BRH$W_RSIZE] = .L2 - BRH$C_LENGTH;
;  1978          3514  2  P[BRH$W_RTYPE] = BRH$K_FILE;
;  1979          3515  2  P[BRH$L_FLAGS] = 0;
;  1980          3516  2  IF .INPUT_FILECHAR[FCH$V_DIRECTORY] THEN P[BRH$V_DIRECTORY] = TRUE;
;  1981          3517  2  P[BRH$L_ADDRESS] = 0;
```

```
;  1982          3518  2 $ASSUME ($BYTEOFFSET (BRH$W_BLOCKFLAGS) + 4, EQL, BRH$K_LENGTH);
;  1983          3519  2 (P[BRH$W_BLOCKFLAGS])<0,32> = 0;
;  1984          3520  2 P = .P + BRH$C_LENGTH;
;  1985          3521  2 STORE_(2, BBH$R_LEVEL1);
;  1986          3522  2
;  1987          3523  2
;  1988          3524  2 ! Store attributes.
;  1989          3525  2 !
;  1990          3526  2 ATV_('FILENAME',          .INPUT_NAMEDESC[DSC$W_LENGTH], .INPUT_NAMEDESC[DSC$A_POINTER]);
;  1991          3527  2 ATT_('STRUCLEV',          .INPUT_HDR[FH2$W_STRUCLEV]);
;  1992          3528  2 ATT_('FID',               INPUT_NAM[NAM$W_FID]);
;  1993          3529  2 ATT_('FILESIZE',          .INPUT_STATBLK[SBK$L_FILESIZE]);
;  1994          3530  2 ATT_('UIC',               .INPUT_FILEOWNER);
;  1995          3531  2 ATV_('RECATTR',          BSA$S_RECATTR, INPUT_RECATTR);
;  1996          3532  2 IF .INPUT_HDR[FH2$B_STRUCLEV] EQL 2
;  1997          3533  2 THEN
;  1998          3534  3     BEGIN
;  1999          3535  3     ATT_('BACKLINK',     INPUT_HDR[FH2$W_BACKLINK]);
;  2000          3536  3     ATT_('FPRO',         .INPUT_HDR[FH2$W_FILEPROT]);
;  2001          3537  3     ATT_('RPRO',         .INPUT_HDR[FH2$W_RECPROT]);
;  2002          3538  3     ATT_('ACLEVEL',      .INPUT_HDR[FH2$B_ACC_MODE]);
;  2003          3539  3     ATT_('UCHAR',        .INPUT_HDR[FH2$L_FILECHAR]);
;  2004          3540  3     ATT_('VERLIMIT',     .DIR_VERLIMIT);
;  2005          3541  3     ATT_('JNL_FLAGS',    .INPUT_HDR[FH2$W_JOURNAL]);
;  2006          3542  3     IF .INPUT_HDR[FH2$B_IDOFFSET] GEQU ($BYTEOFFSET(FH2$L_HIGHWATER)+4)/2
;  2007          3543  3     THEN
;  2008          3544  3         ATT_('HIGHWATER', .INPUT_HDR[FH2$L_HIGHWATER]);
;  2009          3545  3     IF .IDLEN GEQU $BYTEOFFSET(FI2$W_REVISION)+2
;  2010          3546  3     THEN
;  2011          3547  3         ATT_('REVISION',.INPUT_IDENT[FI2$W_REVISION]);
;  2012          3548  3     IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_CREDATE)+8
;  2013          3549  3     THEN
;  2014          3550  3         ATT_('CREDATE', INPUT_CREDATE);
;  2015          3551  3     IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_REVDATE)+8
;  2016          3552  3     THEN
;  2017          3553  3         ATT_('REVDATE', INPUT_REVDATE);
;  2018          3554  3     IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_EXPDATE)+8
;  2019          3555  3     THEN
;  2020          3556  3         ATT_('EXPDATE', INPUT_EXPDATE);
;  2021          3557  3     IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_BAKDATE)+8
;  2022          3558  3     THEN
;  2023          3559  3         IF .QUAL[QUAL_RECO]
;  2024          3560  3         THEN
;  2025          3561  4             ATT_('BAKDATE', JPI_DATE)
;  2026          3562  3         ELSE
;  2027          3563  3             ATT_('BAKDATE', INPUT_BAKDATE);
;  2028          3564  3     END
;  2029          3565  2 ELSE
;  2030          3566  3     BEGIN
;  2031          3567  3     ATT_('FPRO',          .INPUT_HDR[FH1$W_FILEPROT]);
;  2032          3568  3     ATT_('UCHAR',         .INPUT_FILECHAR AND NOT FCH$M_DIRECTORY);
;  2033          3569  3     ATT_('REVISION',     .INPUT_IDENT[FI1$W_REVISION]);
;  2034          3570  3     ATT_('CREDATE',      INPUT_CREDATE);
;  2035          3571  3     ATT_('REVDATE',      INPUT_REVDATE);
;  2036          3572  3     ATT_('EXPDATE',      INPUT_EXPDATE);
;  2037          3573  2     END;
;  2038          3574  2
```

```
;  2039     3575  2
;  2040     3576  2 IF .BOOTVBN NEQ 0
;  2041     3577  2 THEN
;  2042     3578  2     ATT_('BOOTVBN', .BOOTVBN);
;  2043     3579  2
;  2044     3580  2
;  2045     3581  2 IF .PLACEMENT
;  2046     3582  2 THEN
;  2047     3583  3     BEGIN
;  2048     3584  3     STORE_(2, .INPUT_PLACE_LEN);
;  2049     3585  3     STORE_(2, BSA$K_PLACEMENT);
;  2050     3586  3     DUMP_PLC_DATA(.P);
;  2051     3587  3     P = .P + .INPUT_PLACE_LEN;
;  2052     3588  2     END;
;  2053     3589  2
;  2054     3590  2
;  2055     3591  2 IF .INPUT_NAM[NAM$W_DID_NUM] NEQ 0
;  2056     3592  2 THEN
;  2057     3593  3     BEGIN
;  2058     3594  3     ATT_('DIR_UIC', .DIR_SP[D_UIC]);
;  2059     3595  3     ATT_('DIR_FPRO', .DIR_SP[D_FPRO]);
;  2060     3596  3     ATT_('DIR_VERLIM', .DIR_SP[D_VERLIM]);
;  2061     3597  2     END;
;  2062     3598  2
;  2063     3599  2
;  2064     3600  2 IF .DIR_STATUS NEQ 0
;  2065     3601  2 THEN
;  2066     3602  2     ATT_('DIR_STATUS', .DIR_STATUS);
;  2067     3603  2
;  2068     3604  2
;  2069     3605  2 IF .SAVE_ACL_PRI
;  2070     3606  2 THEN
;  2071     3607  2     ATV_ ('ACLSEGMENT', .ACL_LENGTH, .ACL_BUFFER);
;  2072     3608  2
;  2073     3609  2
;  2074     3610  2 ! Pad the primary file attribute record with zeros.
;  2075     3611  2 !
;  2076     3612  2 CH$FILL(0, .L2 - .L1, .P);
;  2077     3613  2
;  2078     3614  2
;  2079     3615  2 ! If the ACL should be saved in a file attribute extension record, do it now.
;  2080     3616  2 !
;  2081     3617  2 IF .SAVE_ACL_EXT
;  2082     3618  2 THEN
;  2083     3619  3     BEGIN
;  2084     3620  3     FIB_DESC[DSC$W_LENGTH] = FIB$C_LENGTH;
;  2085     3621  3     FIB_DESC[DSC$A_POINTER] = FIB_ACL;
;  2086     3622  3     CH$FILL (0, FIB$C_LENGTH, FIB_ACL);
;  2087     3623  3     FIB_ACL[FIB$W_FID_NUM] = .INPUT_NAM[NAM$W_FID_NUM];
;  2088     3624  3     FIB_ACL[FIB$W_FID_SEQ] = .INPUT_NAM[NAM$W_FID_SEQ];
;  2089     3625  3     FIB_ACL[FIB$W_FID_RVN] = .INPUT_NAM[NAM$W_FID_RVN];
;  2090     3626  3
;  2091     3627  3 ! Allocate a buffer to be used in building the extension record.
;  2092     3628  3 !
;  2093     3629  3     ACL_BUFFER = GET_VM (MAX_RECORD);
;  2094     3630  3
;  2095     3631  3 ! Now loop, getting the file's entire ACL.  (This is a loop, because the
```

```
;  2096          3632  3 ! file's ACL is unbounded.)
;  2097          3633  3 !
;  2098          3634  3     WHILE 1
;  2099          3635  3     DO
;  2100          3636  4         BEGIN
;  2101          3637  4         CH$FILL (0, MAX_RECORD, .ACL_BUFFER);
;  2102          3638  4         ATR_DESC[0,0,16,0] = ATR$S_READACL;
;  2103          3639  4         ATR_DESC[2,0,16,0] = ATR$C_READACL;
;  2104          3640  4         ATR_DESC[4,0,32,0] = .ACL_BUFFER;
;  2105          3641  4         ATR_DESC[8,0,32,0] = 0;
;  2106          3642  4
;  2107        P 3643  4         STATUS = $QIOW (CHAN = .INPUT_CHAN,     ! Read the file's ACL
;  2108        P 3644  4                         FUNC = IO$_ACCESS,
;  2109        P 3645  4                         IOSB = IOSB,
;  2110        P 3646  4                         P1 = FIB_DESC,
;  2111          3647  4                         P5 = ATR_DESC);
;  2112          3648  4         IF .STATUS THEN STATUS = .IOSB[0];
;  2113          3649  4         IF .STATUS THEN STATUS = .FIB_ACL[FIB$L_ACL_STATUS];
;  2114          3650  4         IF NOT .STATUS
;  2115          3651  4         THEN
;  2116          3652  5             BEGIN
;  2117          3653  5             IF .STATUS NEQ SS$_ACLEMPTY
;  2118          3654  5             AND .STATUS NEQ SS$_NOMOREACE
;  2119          3655  5             THEN FILE_ERROR (
;  2120          3656  5                     BACKUP$_READERR + STS$K_WARNING,
;  2121          3657  5                     .INPUT_FAB,
;  2122          3658  5                     .STATUS);
;  2123          3659  5             FREE_VM (MAX_RECORD, .ACL_BUFFER);
;  2124          3660  5             RETURN;
;  2125          3661  4             END;
;  2126          3662  4
;  2127          3663  4         ! Now determine the end of the current ACL segment.
;  2128          3664  4         !
;  2129          3665  4         P = .ACL_BUFFER;
;  2130          3666  4         ACL_SEGMENT_SIZE = 0;
;  2131          3667  4         UNTIL .P[ACE$B_SIZE] EQL 0
;  2132          3668  4         DO
;  2133          3669  5             BEGIN
;  2134          3670  5             ACL_SEGMENT_SIZE = .ACL_SEGMENT_SIZE + .P[ACE$B_SIZE];
;  2135          3671  5             P = .P + .P[ACE$B_SIZE];
;  2136          3672  5             IF .P GEQU .ACL_BUFFER + MAX_RECORD THEN EXITLOOP;
;  2137          3673  4             END;
;  2138          3674  4
;  2139          3675  4         ! Allocate room in the output buffer.
;  2140          3676  4         !
;  2141          3677  4         L1 = BRH$C_LENGTH + 2 + .ACL_SEGMENT_SIZE;
;  2142          3678  4         L2 = (.L1 + 15) AND NOT 15;
;  2143          3679  4         P = MAKE_SPACE (.L2);
;  2144          3680  4
;  2145          3681  4
;  2146          3682  4         ! Fill in the extension record
;  2147          3683  4         !
;  2148          3684  4         P[BRH$W_RSIZE] = .L2 - BRH$C_LENGTH;
;  2149          3685  4         P[BRH$W_RTYPE] = BRH$K_FILE_EXT;
;  2150          3686  4         P[BRH$L_FLAGS] = 0;
;  2151          3687  4         P[BRH$L_ADDRESS] = 0;
;  2152          3688  4         $ASSUME ($BYTEOFFSET (BRH$W_BLOCKFLAGS) + 4, EQL, BRH$K_LENGTH);
```

```
;  2153          3689  4          (P[BRH$W_BLOCKFLAGS])<0,32> = 0;
;  2154          3690  4          P = .P + BRH$C_LENGTH;
;  2155          3691  4          STORE_ (2, BBH$K_LEVEL1);
;  2156          3692  4          ATV_ ?'ACLSEGMENT', .ACL_SEGMENT_SIZE, .ACL_BUFFER);
;  2157          3693  4          CH$FILL (0, .L2 - .L1, .P);
;  2158          3694  3          END;
;  2159          3695  3      FREE_VM (.ACL_LENGTH, .ACL_BUFFER);
;  2160          3696  2      END;
;  2161          3697  1 END;


                                31  38  52  49  44  2E  009FF P.AAB:  .ASCII  \.DIR;1\                                  ;


                                                    0FFC 00000 FILE_ATTRIBUTES:
                                                               .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 3348
                                                    5E    90   AE  9E 00002          MOVAB   -112(SP), SP
                                                    5A    04   AC  D0 00006          MOVL    INPUT_HDR, R10                ; 3411
                                                    5B         6A  9A 0000A          MOVZBL  (R10), R11
                                                           6A4B 3F 0000D          PUSHAW  (R10)[R11]
                                                    50    01   AA  9A 00010          MOVZBL  1(R10), R0                    ; 3412
                                                    50         5B  C2 00014          SUBL2   R11, R0
                                          7E        50         01  78 00017          ASHL    #1, R0, IDLEN
                                                    56 00000000' EF 3C 0001B          MOVZWL  INPUT_NAMEDESC, L1            ; 3417
                                                    56    68   A6  9E 00022          MOVAB   104(R6), L1
                                                               7E  D4 00026          CLRL    -(SP)                         ; 3418
                                                    02    07   AA  91 00028          CMPB    7(R10), #2
                                                               3C  12 0002C          BNEQ    6$
                                                               6E  D6 0002E          INCL    (SP)
                                                    56         21  C0 00030          ADDL2   #33, L1                       ; 3422
                                                    28         5B  91 00033          CMPB    R11, #40                      ; 3423
                                                               03  1F 00036          BLSSU   1$
                                                    56         08  C0 00038          ADDL2   #8, L1                        ; 3424
                                                    16    04   AE  D1 0003B 1$:      CMPL    IDLEN, #22                    ; 3425
                                                               03  1F 0003F          BLSSU   2$
                                                    56         06  C0 00041          ADDL2   #6, L1                        ; 3426
                                                    1E    04   AE  D1 00044 2$:      CMPL    IDLEN, #30                    ; 3427
                                                               03  1F 00048          BLSSU   3$
                                                    56         0C  C0 0004A          ADDL2   #12, L1                       ; 3428
                                                    26    04   AE  D1 0004D 3$:      CMPL    IDLEN, #38                    ; 3429
                                                               03  1F 00051          BLSSU   4$
                                                    56         0C  C0 00053          ADDL2   #12, L1                       ; 3430
                                                    2E    04   AE  D1 00056 4$:      CMPL    IDLEN, #46                    ; 3431
                                                               03  1F 0005A          BLSSU   5$
                                                    56         0C  C0 0005C          ADDL2   #12, L1                       ; 3432
                                                    36    04   AE  D1 0005F 5$:      CMPL    IDLEN, #54                    ; 3433
                                                               41  1F 00063          BLSSU   8$
                                                    56         0C  C0 00065          ADDL2   #12, L1                       ; 3434
                                                               3C  11 00068          BRB     8$                            ; 3418
                                                    56         2A  C0 0006A 6$:      ADDL2   #42, L1                       ; 3438
00000000'  FF 00000000'  EF       89    AF                    06  39 0006D          MATCHC  #6, P.AAB, INPUT_NAMEDESC, -  ; 3447
                                                                                             @INPUT_NAMEDESC+4
                                                               03  13 0007B          BEQL    7$
                                                    53         06  D0 0007D          MOVL    #6, R3
                                                    53         06  C2 00080 7$:      SUBL2   #6, R3
```

```
                                21   13 00083         BEQL    8$
              01 00000000'      EF   91 00085         CMPB    INPUT_RECATTR, #1                         ; 3448
                                18   12 0008C         BNEQ    8$
                 00000000'      EF   95 0008E         TSTB    INPUT_RECATTR+1                           ; 3449
                                10   12 00094         BNEQ    8$
              10 00000000'      EF   B1 00096         CMPW    INPUT_RECATTR+2, #16                      ; 3450
                                07   12 0009D         BNEQ    8$
     00000000'  EF              20   88 0009F         BISB2   #32, INPUT_FILECHAR+1                     ; 3452
                                57   D4 000A6 8$:     CLRL    BOOTVBN                                   ; 3456
  7F 00000000'  EF              03   E1 000A8         BBC     #3, QUAL+10, 9$                           ; 3457
              51 00000000'      EF   D0 000B0         MOVL    INPUT_STATBLK, R1                         ; 3460
                                76   13 000B7         BEQL    9$
              50 00000000'      EF   D0 000B9         MOVL    INPUT_NAM, R0                             ; 3461
              50          28    A0   9A 000C0         MOVZBL  40(R0), R0
              50 00000000'FF40       DE 000C4         MOVAL   @FAST_BOOT_LBN[R0], R0
     FFFFFFFF   8F        FC    A0   D1 000CC         CMPL    -4(R0), #-1
                                59   13 000D4         BEQL    9$
              50 00000000'      EF   D0 000D6         MOVL    INPUT_NAM, R0                             ; 3462
              50          28    A0   9A 000DD         MOVZBL  40(R0), R0
              50 00000000'FF40       DE 000E1         MOVAL   @FAST_BOOT_LBN[R0], R0
            FC  A0              51   D1 000E9         CMPL    R1, -4(R0)
                                40   1A 000ED         BGTRU   9$
  52          51 00000000'      EF   C1 000EF         ADDL3   INPUT_STATBLK+4, R1, R2                   ; 3463
              50 00000000'      EF   D0 000F7         MOVL    INPUT_NAM, R0
              50          28    A0   9A 000FE         MOVZBL  40(R0), R0
              50 00000000'FF40       DE 00102         MOVAL   @FAST_BOOT_LBN[R0], R0
            FC  A0              52   D1 0010A         CMPL    R2, -4(R0)
                                1F   1B 0010E         BLEQU   9$
              50 00000000'      EF   D0 00110         MOVL    INPUT_NAM, R0                             ; 3466
              50          28    A0   9A 00117         MOVZBL  40(R0), R0
              50 00000000'FF40       DE 0011B         MOVAL   @FAST_BOOT_LBN[R0], R0
  50        FC  A0              51   C3 00123         SUBL3   R1, -4(R0), R0
              57          01    A0   9E 00128         MOVAB   1(R0), BOOTVBN
              56                08   C0 0012C         ADDL2   #8, L1                                    ; 3467
                          0C    AE   D4 0012F 9$:     CLRL    PLACEMENT                                 ; 3471
  2D 00000000'  EF              03   E1 00132         BBC     #3, QUAL+10, 10$                          ; 3473
                 00000000'      EF   B5 0013A         TSTW    INPUT_PLACE_LEN                           ; 3474
                                25   13 00140         BEQL    10$
              50 00000000'      EF   3C 00142         MOVZWL  INPUT_PLACE_LEN, R0                       ; 3475
              50          04 A046    9E 00149         MOVAB   4(R0)[L1], R0
     00000800   8F              50   D1 0014E         CMPL    R0, #2048
                                10   1A 00155         BGTRU   10$
              50 00000000'      EF   3C 00157         MOVZWL  INPUT_PLACE_LEN, R0                       ; 3478
              56          04 A046    9E 0015E         MOVAB   4(R0)[L1], L1
            0C  AE              01   D0 00163         MOVL    #1, PLACEMENT                             ; 3479
              50 00000000'      EF   D0 00167 10$:    MOVL    INPUT_NAM, R0                             ; 3483
                          2A    A0   B5 0016E         TSTW    42(R0)
                                03   13 00171         BEQL    11$
              56                14   C0 00173         ADDL2   #20, L1                                   ; 3485
                 00000000'      EF   95 00176 11$:    TSTB    DIR_STATUS                                ; 3488
                                03   13 0017C         BEQL    12$
              56                05   C0 0017E         ADDL2   #5, L1                                    ; 3490
                          10    AE   7C 00181 12$:    CLRQ    SAVE_ACL_EXT                              ; 3493
              50 00000000'      EF   D0 00184         MOVL    ACL_LENGTH, R0                            ; 3494
                                26   15 0018B         BLEQ    14$
  1E 00000000'  EF              01   E0 0018D         BBS     #1, QUAL+14, 14$                          ; 3495
              1B                6E   E9 00195         BLBC    (SP), 14$                                 ; 3496
```

```
                                50       04 A046   9E 00198          MOVAB   4(R0)[L1], R0                                ; 3499
                     00000800   8F            50   D1 0019D          CMPL    R0, #2048                                    ;
                                                09   1A 001A4          BGTRU   13$                                          ;
                                56              50   D0 001A6          MOVL    R0, L1                                       ; 3502
                           14   AE              01   D0 001A9          MOVL    #1, SAVE_ACL_PRI                             ; 3503
                                                04   11 001AD          BRB     14$                                          ; 3499
                           10   AE              01   D0 001AF 13$:     MOVL    #1, SAVE_ACL_EXT                             ; 3505
                                50       0F     A6   9E 001B3 14$:     MOVAB   15(R6), R0                                   ; 3511
                   58           50              0F   CB 001B7          BICL3   #15, R0, L2                                  ;
                                                58   DD 001BB          PUSHL   L2                                           ; 3512
                         F5D9   CF              01   FB 001BD          CALLS   #1, MAKE_SPACE                               ;
                                59              50   D0 001C2          MOVL    R0, P                                        ;
                   69           58              10   A3 001C5          SUBW3   #16, L2, (P)                                 ; 3513
                           02   A9              03   B0 001C9          MOVW    #3, 2(P)                                     ; 3514
                                        04      A9   D4 001CD          CLRL    4(P)                                         ; 3515
                   04 00000000' EF              05   E1 001D0          BBC     #5, INPUT_FILECHAR+1, 15$                    ; 3516
                           04   A9              02   88 001D8          BISB2   #2, 4(P)                                     ;
                                        08      A9   7C 001DC 15$:     CLRQ    8(P)                                         ; 3517
                                59              10   C0 001DF          ADDL2   #16, P                                       ; 3520
                                89         0101 8F   B0 001E2          MOVW    #257, (P)+                                   ; 3521
                                89 00000000' EF   B0 001E7          MOVW    INPUT_NAMEDESC, (P)+                         ; 3526
                                89           2A   B0 001EE          MOVW    #42, (P)+                                    ;
                   69 00000000' FF 00000000' EF   28 001F1          MOVC3   INPUT_NAMEDESC, @INPUT_NAMEDESC+4, (P)       ;
                                59              53   D0 001FD          MOVL    R3, P                                        ;
                                89 002B0002     8F   D0 00200          MOVL    #2818050, (P)+                               ; 3527
                                89           06 AA   B0 00207          MOVW    6(R10), (P)+                                 ;
                                89 002C0006     8F   D0 0020B          MOVL    #2883590, (P)+                               ; 3528
                                50 00000000'    EF   D0 00212          MOVL    INPUT_NAM, R0                                ;
                                89           24 A0   D0 00219          MOVL    36(R0), (P)+                                 ;
                                89           28 A0   B0 0021D          MOVW    40(R0), (P)+                                 ;
                                89 002E0004     8F   D0 00221          MOVL    #3014660, (P)+                               ; 3529
                                89 00000000'    EF   D0 00228          MOVL    INPUT_STATBLK+4, (P)+                        ;
                                89 002F0004     8F   D0 0022F          MOVL    #3080796, (P)+                               ; 3530
                                89 00000000'    EF   D0 00236          MOVL    INPUT_FILEOWNER, (P)+                        ;
                                89 00340020     8F   D0 0023D          MOVL    #3407904, (P)+                               ; 3531
                   69 00000000' EF              20   28 00244          MOVC3   #32, INPUT_RECATTR, (P)                      ;
                                59              53   D0 0024C          MOVL    R3, P                                        ;
                                03              6E   E8 0024F          BLBS    (SP), 16$                                    ; 3535
                                             00F4    31 00252          BRW     24$                                          ;
                                89 002D0006     8F   D0 00255 16$:     MOVL    #2949126, (P)+                               ;
                                89           42 AA   D0 0025C          MOVL    66(R10), (P)+                                ;
                                89           46 AA   B0 00260          MOVW    70(R10), (P)+                                ;
                                89 00300002     8F   D0 00264          MOVL    #3145750, (P)+                               ; 3536
                                89           40 AA   B0 0026B          MOVW    64(R10), (P)+                                ;
                                89 00310002     8F   D0 0026F          MOVL    #3211286, (P)+                               ; 3537
                                89           38 AA   B0 00276          MOVW    56(R10), (P)+                                ;
                                89 00320001     8F   D0 0027A          MOVL    #3276801, (P)+                               ; 3538
                                89           3B AA   90 00281          MOVB    59(R10), (P)+                                ;
                                89 00330004     8F   D0 00285          MOVL    #3342340, (P)+                               ; 3539
                                89           34 AA   D0 0028C          MOVL    52(R10), (P)+                                ;
                                89 004B0002     8F   D0 00290          MOVL    #4915202, (P)+                               ; 3540
                                89 00000000'    EF   B0 00297          MOVW    DIR_VERLIMIT, (P)+                           ;
                                89 00500002     8F   D0 0029E          MOVL    #5242882, (P)+                               ; 3541
                                89           48 AA   B0 002A5          MOVW    72(R10), (P)+                                ;
                                28              5B   91 002A9          CMPB    R11, #40                                     ; 3542
                                                0B   1F 002AC          BLSSU   17$                                          ;
                                89 004F0004     8F   D0 002AE          MOVL    #5177348, (P)+                               ; 3544
```

```
                    89        4C    AA   D0 002B5          MOVL    76(R10), (P)+
                    16        04    AE   D1 002B9 17$:     CMPL    IDLEN, #22                                  ; 3545
                                    0F   1F 002BD          BLSSU   18$
                    89 00350002     8F   D0 002BF          MOVL    #3473410, (P)+                              ; 3547
          50        08    AE              14   C1 002C6    ADDL3   #20, INPUT_IDENT, R0
                    89              60   B0 002CB          MOVW    (R0), (P)+
                    1E        04    AE   D1 002CE 18$:     CMPL    IDLEN, #30                                  ; 3548
                                    0E   1F 002D2          BLSSU   19$
                    89 00360008     8F   D0 002D4          MOVL    #3538952, (P)+                              ; 3550
                    89 00000000'    EF   7D 002DB          MOVQ    INPUT_CREDATE, (P)+
                    26        04    AE   D1 002E2 19$:     CMPL    IDLEN, #38                                  ; 3551
                                    0E   1F 002E6          BLSSU   20$
                    89 00370008     8F   D0 002E8          MOVL    #3604488, (P)+                              ; 3553
                    89 00000000'    EF   7D 002EF          MOVQ    INPUT_REVDATE, (P)+
                    2E        04    AE   D1 002F6 20$:     CMPL    IDLEN, #46                                  ; 3554
                                    0E   1F 002FA          BLSSU   21$
                    89 00380008     8F   D0 002FC          MOVL    #3670024, (P)+                              ; 3556
                    89 00000000'    EF   7D 00303          MOVQ    INPUT_EXPDATE, (P)+
                    36        04    AE   D1 0030A 21$:     CMPL    IDLEN, #54                                  ; 3557
                                    03   1E 0030E          BGEQU   22$
                                0097    31   00310          BRW     26$
          17 00000000'    EF              06   E1 00313 22$:     BBC     #6, QUAL+12, 23$                    ; 3559
                    89 00390008     8F   D0 0031B          MOVL    #3735560, (P)+                              ; 3561
                    89 00000000'    EF   D0 00322          MOVL    JPI_DATE, (P)+
                    69 00000000'    EF   D0 00329          MOVL    JPI_DATE+4, (P)
                                    75   11 00330          BRB     25$
                    89 00390008     8F   D0 00332 23$:     MOVL    #3735560, (P)+                              ; 3563
                    89 00000000'    EF   D0 00339          MOVL    INPUT_BAKDATE, (P)+
                    69 00000000'    EF   D0 00340          MOVL    INPUT_BAKDATE+4, (P)
                                    5E   11 00347          BRB     25$
                    89 00300002     8F   D0 00349 24$:     MOVL    #3145730, (P)+                              ; 3567
                    89        0A    AA   B0 00350          MOVW    10(R10), (P)+
                    89 00330004     8F   D0 00354          MOVL    #3342340, (P)+                              ; 3568
    89 00000000'    EF 00002000     8F   CB 0035B          BICL3   #8192, INPUT_FILECHAR, (P)+
                    89 00350002     8F   D0 00367          MOVL    #3473410, (P)+                              ; 3569
          50        08    AE              0A   C1 0036E    ADDL3   #10, INPUT_IDENT, R0
                    89              60   B0 00373          MOVW    (R0), (P)+
                    89 00360008     8F   D0 00376          MOVL    #3538952, (P)+                              ; 3570
                    89 00000000'    EF   7D 0037D          MOVQ    INPUT_CREDATE, (P)+
                    89 00370008     8F   D0 00384          MOVL    #3604488, (P)+                              ; 3571
                    89 00000000'    EF   7D 0038B          MOVQ    INPUT_REVDATE, (P)+
                    89 00380008     8F   D0 00392          MOVL    #3670024, (P)+                              ; 3572
                    89 00000000'    EF   D0 00399          MOVL    INPUT_EXPDATE, (P)+
                    69 00000000'    EF   D0 003A0          MOVL    INPUT_EXPDATE+4, (P)
                    59                   04   C0 003A7 25$:     ADDL2   #4, P
                                         57   D5 003AA 26$:     TSTL    BOOTVBN                                 ; 3576
                                         0A   13 003AC          BEQL    27$
                    89 00450004          8F   D0 003AE          MOVL    #4521988, (P)+                          ; 3578
                    89                   57   D0 003B5          MOVL    BOOTVBN, (P)+
                    1C              0C   AE   E9 003B8 27$:     BLBC    PLACEMENT, 28$                          ; 3581
                    89 00000000'    EF   B0 003BC          MOVW    INPUT_PLACE_LEN, (P)+                       ; 3584
                    89              46   8F   9B 003C3          MOVZBW  #70, (P)+                               ; 3585
                                         59   DD 003C7          PUSHL   P                                       ; 3586
                FBFE    CF                   01   FB 003C9          CALLS   #1, DUMP_PLC_DATA
                    50 00000000'    EF   3C 003CE          MOVZWL  INPUT_PLACE_LEN, R0                         ; 3587
                    59                   50   C0 003D5          ADDL2   R0, P
                    57 00000000'    EF   D0 003D8 28$:     MOVL    INPUT_NAM, R7                               ; 3591
```

```
                                        2A   A7  B5 003DF              TSTW    42(R7)                                          ;
                                             28  13 003E2              BEQL    29$                                             ;
                                     89 00470004  8F  D0 003E4              MOVL    #4653060, (P)+                          ; 3594
                                     50 00000000' EF  D0 003EB              MOVL    DIR_SP, R0                              ;
                                     89       3C  A0  D0 003F2              MOVL    60(R0), (P)+                            ;
                                     89 00480002  8F  D0 003F6              MOVL    #4718594, (P)+                          ; 3595
                                     89       3A  A0  B0 003FD              MOVW    58(R0), (P)+                            ;
                                     89 004A0002  8F  D0 00401              MOVL    #4849666, (P)+                          ; 3596
                                     89       40  A0  B0 00408              MOVW    64(R0), (P)+                            ;
                                     50 00000000' EF  9A 0040C 29$:         MOVZBL  DIR_STATUS, R0                          ; 3600
                                                  0A  13 00413              BEQL    30$                                     ;
                                     89 00490001  8F  D0 00415              MOVL    #4784129, (P)+                          ; 3602
                                     89           50  90 0041C              MOVB    R0, (P)+                                ;
                                     1A       14  AE  E9 0041F 30$:         BLBC    SAVE_ACL_PRI, 31$                       ; 3605
                                     89 00000000' EF  B0 00423              MOVW    ACL_LENGTH, (P)+                        ; 3607
                                     89       4E  8F  9B 0042A              MOVZBW  #78, (P)+                               ;
                 69 00000000'  FF 00000000'  EF  28 0042E                   MOVC3   ACL_LENGTH, @ACL_BUFFER, (P)            ;
                               59            53  D0 0043A                   MOVL    R3, P                                   ;
                 50            58            56  C3 0043D 31$:              SUBL3   L1, L2, R0                              ; 3612
           50    00            6E            00  2C 00441                   MOVC5   #0, (SP), #0, R0, (P)                   ;
                                               69  00446                   ;                                               ;
                                     01       10  AE  E8 00447              BLBS    SAVE_ACL_EXT, 32$                       ; 3617
                                                  04  0044B              RET                                             ;
                                 60  AE       40  8F  9B 0044C 32$:         MOVZBW  #64, FIB_DESC                           ; 3620
                                 64  AE       20  AE  9E 00451              MOVAB   FIB_ACL, FIB_DESC+4                     ; 3621
      0040 8F    00               6E            00  2C 00456              MOVC5   #0, (SP), #0, #64, FIB_ACL              ; 3622
                                               20  AE  00459              ;                                               ;
                                 24  AE       24  A7  D0 0045F              MOVL    36(R7), FIB_ACL+4                       ; 3623
                                 28  AE       28  A7  B0 00464              MOVW    40(R7), FIB_ACL+8                       ; 3625
                                 7E       0800  8F  3C 00469              MOVZWL  #2048, -(SP)                            ; 3629
                    00000000G    00            01  FB 0046E              CALLS   #1, GET_VM                              ;
                    00000000'    EF            50  D0 00475              MOVL    R0, ACL_BUFFER                          ;
                                 5B 00000000' EF  D0 0047C              MOVL    ACL_BUFFER, R11                         ; 3637
  0800 8F           00            6E            00  2C 00483 33$:         MOVC5   #0, (SP), #0, #2048, (R11)              ;
                                               6B  0048A                  ;                                               ;
                                 68  AE 00250200  8F  D0 0048B              MOVL    #2425344, ATR_DESC                      ; 3638
                                 6C  AE           5B  D0 00493              MOVL    R11, ATR_DESC+4                         ; 3640
                                          70  AE  D4  00497              CLRL    ATR_DESC+8                              ; 3641
                                              7E  D4  0049A              CLRL    -(SP)                                   ; 3647
                                          6C  AE  9F  0049C              PUSHAB  ATR_DESC                                ;
                                              7E  7C  0049F              CLRQ    -(SP)                                   ;
                                              7E  D4  004A1              CLRL    -(SP)                                   ;
                                          74  AE  9F  004A3              PUSHAB  FIB_DESC                                ;
                                              7E  7C  004A6              CLRQ    -(SP)                                   ;
                                          38  AE  9F  004A8              PUSHAB  IOSB                                    ;
                                              32  DD  004AB              PUSHL   #50                                     ;
                                 00000000'    EF  DD  004AD              PUSHL   INPUT_CHAN                              ;
                                              7E  D4  004B3              CLRL    -(SP)                                   ;
                    00000000G    00           0C  FB  004B5              CALLS   #12, SYS$QIOW                           ;
                                 5A           50  D0  004BC              MOVL    R0, STATUS                              ;
                                 0E           5A  E9  004BF              BLBC    STATUS, 34$                             ; 3648
                                 5A       18  AE  3C  004C2              MOVZWL  IOSB, STATUS                            ;
                                 07           5A  E9  004C6              BLBC    STATUS, 34$                             ; 3649
                                 5A       54  AE  D0  004C9              MOVL    FIB_ACL+52, STATUS                      ;
                                 34           5A  E8  004CD              BLBS    STATUS, 36$                             ; 3650
                    000009D0     8F           5A  D1  004D0 34$:         CMPL    STATUS, #2512                           ; 3653
                                              1E  13  004D7              BEQL    35$                                     ;
```

```
                 000009E0   8F                5A  D1 004D9          CMPL    STATUS, #2528                                   ; 3654
                                              15  13 004E0          BEQL    35$                                             ;
                                              5A  DD 004E2          PUSHL   STATUS                                          ; 3658
                                00000000'     EF  DD 004E4          PUSHL   INPUT_FAB                                       ; 3657
                                00000000G     8F  DD 004EA          PUSHL   #BACKUP$_READERR                                ; 3656
                 00000000G      00            03  FB 004F0          CALLS   #3, FILE_ERROR                                  ;
                                00000000'     EF  DD 004F7 35$:      PUSHL   ACL_BUFFER                                      ; 3659
                                7E      0800  8F  3C 004FD          MOVZWL  #2048, -(SP)                                    ;
                                              72  11 00502          BRB     39$                                             ;
                                59 00000000'  EF  D0 00504 36$:      MOVL    ACL_BUFFER, P                                   ; 3665
                                              57  D4 0050B          CLRL    ACL_SEGMENT_SIZE                                ; 3666
                50 00000000'    EF 00000800   8F  C1 0050D          ADDL3   #2048, ACL_BUFFER, R0                           ; 3672
                                              69  95 00519 37$:      TSTB    (P)                                             ; 3667
                                              11  13 0051B          BEQL    38$                                             ;
                                51            69  9A 0051D          MOVZBL  (P), R1                                         ; 3670
                                57            51  C0 00520          ADDL2   R1, ACL_SEGMENT_SIZE                            ;
                                51            69  9A 00523          MOVZBL  (P), R1                                         ; 3671
                                59            51  C0 00526          ADDL2   R1, P                                           ;
                                50            59  D1 00529          CMPL    P, R0                                           ; 3672
                                              EB  1F 0052C          BLSSU   37$                                             ;
                                56      12    A7  9E 0052E 38$:      MOVAB   18(R7), L1                                      ; 3677
                                50      0F    A6  9E 00532          MOVAB   15(R6), R0                                      ; 3678
                58              50            0F  CB 00536          BICL3   #15, R0, L2                                     ;
                                              58  DD 0053A          PUSHL   L2                                              ; 3679
                      F25A      CF            01  FB 0053C          CALLS   #1, MAKE_SPACE                                  ;
                                59            50  D0 00541          MOVL    R0, P                                           ;
                89              58            10  A3 00544          SUBW3   #16, L2, (P)+                                   ; 3684
                                89            08  B0 00548          MOVW    #8, (P)+                                        ; 3685
                                              89  7C 0054B          CLRQ    (P)+                                            ; 3686
                                              89  D4 0054D          CLRL    (P)+                                            ; 3689
                                89      0101  8F  B0 0054F          MOVW    #257, (P)+                                      ; 3691
                                89            57  B0 00554          MOVW    ACL_SEGMENT_SIZE, (P)+                          ; 3692
                                89      4E    8F  9B 00557          MOVZBW  #78, (P)+                                       ;
                                5B 00000000'  EF  D0 0055B          MOVL    ACL_BUFFER, R11                                 ;
                    69          6B            57  28 00562          MOVC3   ACL_SEGMENT_SIZE, (R11), (P)                    ;
                                59            53  D0 00566          MOVL    R3, P                                           ;
                    50          58            56  C3 00569          SUBL3   L1, L2, R0                                      ; 3693
          50        00          6E            00  2C 0056D          MOVC5   #0, (SP), #0, R0, (P)                           ;
                                              69  00572                                                                   ;
                                          FF0D  31  00573          BRW     33$                                             ; 3634
                      00000000G   00            02  FB 00576 39$:      CALLS   #2, FREE_VM                                     ; 3695
                                                04  0057D          RET                                                     ; 3697

; Routine Size:  1406 bytes,    Routine Base:  CODE + 0A05
```

```
;  2163          3698  1 %SBTTL 'VOLUME_ATTRIBUTES - format volume attribute record'
;  2164          3699  1 GLOBAL ROUTINE VOLUME_ATTRIBUTES(INPUT_HOME,INPUT_BOOT,INPUT_SCB,INPUT_MAXFILNUM): NOVALUE=
;  2165          3700  1
;  2166          3701  1 !++
;  2167          3702  1 !
;  2168          3703  1 ! FUNCTIONAL DESCRIPTION:
;  2169          3704  1 !       This routine generates a volume attribute record.
;  2170          3705  1 !
;  2171          3706  1 ! INPUT PARAMETERS:
;  2172          3707  1 !       INPUT_HOME      - Pointer to home block.
;  2173          3708  1 !       INPUT_BOOT      - Pointer to boot block.
;  2174          3709  1 !       INPUT_SCB       - Pointer to storage control block.
;  2175          3710  1 !       INPUT_MAXFILNUM - Highest file number in use.
;  2176          3711  1 !
;  2177          3712  1 ! IMPLICIT INPUTS:
;  2178          3713  1 !       NONE
;  2179          3714  1 !
;  2180          3715  1 ! OUTPUT PARAMETERS:
;  2181          3716  1 !       NONE
;  2182          3717  1 !
;  2183          3718  1 ! IMPLICIT OUTPUTS:
;  2184          3719  1 !       NONE
;  2185          3720  1 !
;  2186          3721  1 ! ROUTINE VALUE:
;  2187          3722  1 !       NONE
;  2188          3723  1 !
;  2189          3724  1 ! SIDE EFFECTS:
;  2190          3725  1 !       NONE
;  2191          3726  1 !
;  2192          3727  1 !--
;  2193          3728  1
;  2194          3729  2 BEGIN
;  2195          3730  2 MAP
;  2196          3731  2         INPUT_HOME:     REF BBLOCK,     ! Pointer to home block
;  2197          3732  2         INPUT_BOOT:     REF BBLOCK,     ! Pointer to boot block
;  2198          3733  2         INPUT_SCB:      REF BBLOCK;     ! Pointer to storage control block
;  2199          3734  2 LOCAL
;  2200          3735  2         BOOT_LENGTH,                    ! Boot block length less trailing zeros
;  2201          3736  2         L1,                             ! Unrounded length
;  2202          3737  2         L2,                             ! Rounded length
;  2203          3738  2         P:              REF BBLOCK;     ! Pointer to attribute record
;  2204          3739  2 LITERAL
;  2205          3740  2     FIXED_SIZE =
;  2206          3741  2         BRH$C_LENGTH +          ! record header
;  2207          3742  2         2 +                     ! structure level
;  2208          3743  2         18*4 +                  ! 18 attribute size/type longwords
;  2209          3744  2         BSA$S_VOLSTRUCT +       !  1. volume structure level
;  2210          3745  2         BSA$S_VOLNAME +         !  2. volume label
;  2211          3746  2         BSA$S_OWNERNAME +       !  3. volume owner name
;  2212          3747  2         BSA$S_FORMAT +          !  4. volume file format name
;  2213          3748  2         BSA$S_VOLOWNER +        !  5. volume owner UIC
;  2214          3749  2         BSA$S_PROTECT +         !  6. volume protection mask
;  2215          3750  2         BSA$S_FILEPROT +        !  7. volume default file protection
;  2216          3751  2         BSA$S_VOLCHAR +         !  8. volume characteristics bits
;  2217          3752  2         BSA$S_VOLDATE +         !  9. volume creation date
;  2218          3753  2         BSA$S_WINDOW +          ! 10. volume default file window size
;  2219          3754  2         BSA$S_LRU_LIM +         ! 11. volume default directory LRU limit
```

```
; 2220          3755  2          BSA$S_EXTEND +          ! 12. volume default file extension size
; 2221          3756  2          BSA$S_CLUSTER +         ! 13. volume cluster factor
; 2222          3757  2          BSA$S_VOLSIZE +         ! 14. volume size in blocks
; 2223          3758  2          BSA$S_MAXFILES +        ! 15. volume maximum number of files
; 2224          3759  2          BSA$S_MAXFILNUM +       ! 16. highest file number in use
; 2225          3760  2          BSA$S_SERIALNUM +       ! 17. volume serial number
; 2226          3761  2          BSA$S_INDEXLBN;         ! 18. index file bitmap starting LBN
; 2227          3762  2
; 2228          3763  2
; 2229          3764  2 ! Generate the backup summary record.  Determine the length of the attribute
; 2230          3765  2 ! record.
; 2231          3766  2 !
; 2232          3767  2 L1 = FIXED_SIZE;
; 2233          3768  2 IF .INPUT_HOME[HM2$B_STRUCLEV] EQL 2
; 2234          3769  2 THEN
; 2235          3770  3     BEGIN
; 2236          3771  3     IF
; 2237          3772  4         (IF .QUAL[QUAL_VOLU]
; 2238          3773  4             THEN .QUAL[QUAL_VOLU_VALUE] EQL .INPUT_HOME[HM2$W_RVN]
; 2239          3774  4             ELSE .INPUT_HOME[HM2$W_RVN] LEQU 1)
; 2240          3775  3     THEN
; 2241          3776  3         BACKUP_SUMMARY();
; 2242          3777  3
; 2243          3778  3     L1 = .L1 + 4*4 + BSA$S_RECPROT + BSA$S_RESFILES + BSA$S_RETAINMIN + BSA$S_RETAINMAX;
; 2244          3779  3     IF .COM_I_SETCOUNT GTRU 1 THEN L1 = .L1 + 4 + BSA$S_RVN;
; 2245          3780  3     END
; 2246          3781  2 ELSE
; 2247          3782  2     BACKUP_SUMMARY();
; 2248          3783  2
; 2249          3784  2
; 2250          3785  2 ! If the volume is bootable, determine the length of the boot program with
; 2251          3786  2 ! trailing zero bytes stripped off.
; 2252          3787  2 !
; 2253          3788  2 IF .INPUT_BOOT[5,0,8,0] EQL 0
; 2254          3789  2 THEN
; 2255          3790  3     BEGIN
; 2256          3791  3     BOOT_LENGTH = 0;
; 2257          3792  3     DECR J FROM 511 TO 0 DO
; 2258          3793  4         BEGIN
; 2259          3794  4         IF .INPUT_BOOT[.J,0,8,0] NEQ 0
; 2260          3795  4         THEN
; 2261          3796  5             BEGIN
; 2262          3797  5             BOOT_LENGTH = .J + 1;
; 2263          3798  5             EXITLOOP;
; 2264          3799  4             END;
; 2265          3800  3         END;
; 2266          3801  3     L1 = .L1 + 4 + .BOOT_LENGTH;
; 2267          3802  2     END;
; 2268          3803  2
; 2269          3804  2
; 2270          3805  2 ! Allocate the attribute record and initialize the record header.
; 2271          3806  2 !
; 2272          3807  2 L2 = (.L1 + 15) AND NOT 15;
; 2273          3808  2 P = MAKE_SPACE(.L2);
; 2274          3809  2 P[BRH$W_RSIZE] = .L2 - BRH$C_LENGTH;
; 2275          3810  2 P[BRH$W_RTYPE] = BRH$K_VOLUME;
; 2276          3811  2 P[BRH$L_FLAGS] = 0;
```

```
; 2277          3812  2 P[BRH$L_ADDRESS] = 0;
; 2278          3813  2 $ASSUME (SBYTEOFFSET (BRH$W_BLOCKFLAGS) + 4, EQL, BRH$K_LENGTH);
; 2279          3814  2 (P[BRH$W_BLOCKFLAGS])<0,32> = 0;
; 2280          3815  2 P = .P + BRH$C_LENGTH;
; 2281          3816  2 STORE_(2, BBH$R_LEVEL1);
; 2282          3817  2
; 2283          3818  2
; 2284          3819  2 ! Store attributes.
; 2285          3820  2 !
; 2286          3821  2 IF .INPUT_HOME[HM2$B_STRUCLEV] EQL 2
; 2287          3822  2 THEN
; 2288          3823  3     BEGIN
; 2289          3824  3     ATT_('VOLSTRUCT',    .INPUT_HOME[HM2$W_STRUCLEV]);
; 2290          3825  3     ATT_('VOLNAME',      INPUT_HOME[HM2$T_VOLNAME]);
; 2291          3826  3     ATT_('OWNERNAME',    INPUT_HOME[HM2$T_OWNERNAME]);
; 2292          3827  3     ATT_('FORMAT',       INPUT_HOME[HM2$T_FORMAT]);
; 2293          3828  3     IF .COM_I_SETCOUNT GTRU 1
; 2294          3829  3     THEN
; 2295          3830  3         ATT_('RVN',      .INPUT_HOME[HM2$W_RVN]);
; 2296          3831  3     ATT_('VOLOWNER',     .INPUT_HOME[HM2$L_VOLOWNER]);
; 2297          3832  3     ATT_('PROTECT',      .INPUT_HOME[HM2$W_PROTECT]);
; 2298          3833  3     ATT_('FILEPROT',     .INPUT_HOME[HM2$W_FILEPROT]);
; 2299          3834  3     ATT_('RECPROT',      .INPUT_HOME[HM2$W_RECPROT]);
; 2300          3835  3     ATT_('VOLCHAR',      .INPUT_HOME[HM2$W_VOLCHAR]);
; 2301          3836  3     ATT_('VOLDATE',      INPUT_HOME[HM2$Q_CREDATE]);
; 2302          3837  3     ATT_('WINDOW',       .INPUT_HOME[HM2$B_WINDOW]);
; 2303          3838  3     ATT_('LRU_LIM',      .INPUT_HOME[HM2$B_LRU_LIM]);
; 2304          3839  3     ATT_('EXTEND',       .INPUT_HOME[HM2$W_EXTEND]);
; 2305          3840  3     ATT_('CLUSTER',      .INPUT_HOME[HM2$W_CLUSTER]);
; 2306          3841  3     ATT_('RESFILES',     .INPUT_HOME[HM2$W_RESFILES]);
; 2307          3842  3     ATT_('VOLSIZE',      .INPUT_SCB[SCB$L_VOLSIZE]);
; 2308          3843  3     ATT_('MAXFILES',     .INPUT_HOME[HM2$L_MAXFILES]);
; 2309          3844  3     ATT_('MAXFILNUM',    .INPUT_MAXFILNUM);
; 2310          3845  3     ATT_('SERIALNUM',    .INPUT_HOME[HM2$L_SERIALNUM]);
; 2311          3846  3     ATT_('INDEXLBN',     .INPUT_HOME[HM2$L_IBMAPLBN]);
; 2312          3847  3     ATT_('RETAINMIN',    INPUT_HOME[HM2$Q_RETAINMIN]);
; 2313          3848  3     ATT_('RETAINMAX',    INPUT_HOME[HM2$Q_RETAINMAX]);
; 2314          3849  3     END
; 2315          3850  2 ELSE
; 2316          3851  3     BEGIN
; 2317          3852  3     LOCAL
; 2318          3853  3         CREDATE:         VECTOR[2],      ! Temp to get creation date
; 2319          3854  3         T;                               ! Temp to assemble UIC
; 2320          3855  3
; 2321          3856  3     ATT_('VOLSTRUCT',    .INPUT_HOME[HM1$W_STRUCLEV]);
; 2322          3857  3     ATT_('VOLNAME',      INPUT_HOME[HM1$T_VOLNAME2]);
; 2323          3858  3     ATT_('OWNERNAME',    INPUT_HOME[HM1$T_OWNERNAME]);
; 2324          3859  3     ATT_('FORMAT',       INPUT_HOME[HM1$T_FORMAT]);
; 2325          3860  3     T< 0,16> = .(INPUT_HOME[HM1$W_VOLOWNER])<0,8>;
; 2326          3861  3     T<16,16> = .(INPUT_HOME[HM1$W_VOLOWNER])<8,8>;
; 2327          3862  3     ATT_('VOLOWNER',     .T);
; 2328          3863  3     ATT_('PROTECT',      .INPUT_HOME[HM1$W_PROTECT]);
; 2329          3864  3     ATT_('FILEPROT',     .INPUT_HOME[HM1$W_FILEPROT]);
; 2330          3865  3     ATT_('VOLCHAR',      .INPUT_HOME[HM2$W_VOLCHAR]);
; 2331          3866  3     FROM_ODS1_DATE(INPUT_HOME[HM1$T_CREDATE], CREDATE);
; 2332          3867  3     ATT_('VOLDATE',      CREDATE);
; 2333          3868  3     ATT_('WINDOW',       .INPUT_HOME[HM1$B_WINDOW]);
```

```
; 2334     3869  3      ATT_('LRU_LIM',      .INPUT_HOME[HM1$B_LRU_LIM]);
; 2335     3870  3      ATT_('EXTEND',       .INPUT_HOME[HM1$B_EXTEND]);
; 2336     3871  3      ATT_('CLUSTER',      .INPUT_HOME[HM1$W_CLUSTER]);
; 2337     3872  3      ATT_('VOLSIZE',     ROT(.(.INPUT_SCB + 4 + .INPUT_SCB[3,0,8,0] * 4), 16));
; 2338     3873  3      ATT_('MAXFILES',     .INPUT_HOME[HM1$W_MAXFILES]);
; 2339     3874  3      ATT_('MAXFILNUM',    .INPUT_MAXFILNUM);
; 2340     3875  3      ATT_('SERIALNUM',    .INPUT_HOME[HM1$L_SERIALNUM]);
; 2341     3876  3      ATT_('INDEXLBN',    ROT(.INPUT_HOME[HM1$L_IBMAPLBN], 16));
; 2342     3877  2      END;
; 2343     3878  2
; 2344     3879  2
; 2345     3880  2  IF .INPUT_BOOT[5,0,8,0] EQL 0
; 2346     3881  2  THEN
; 2347     3882  2      ATV_('BOOTBLOCK',    .BOOT_LENGTH, .INPUT_BOOT);
; 2348     3883  2
; 2349     3884  2
; 2350     3885  2  ! Pad the record with zeros.
; 2351     3886  2  !
; 2352     3887  2  CH$FILL(0, .L2 - .L1, .P);
; 2353     3888  1  END;


                                    0FFC 00000          .ENTRY  VOLUME_ATTRIBUTES, Save R2,R3,R4,R5,R6,R7,- ; 3699
                                                                R8,R9,R10,R11                                   ;
                5B 00000000'   EF  9E 00002             MOVAB   COM_I_SETCOUNT, R11                             ;
                5E             08  C2 00009             SUBL2   #8, SP                                          ;
                56       AC    8F  9A 0000C             MOVZBL  #172, L1                                        ; 3767
                54       04    AC  D0 00010             MOVL    INPUT_HOME, R4                                  ; 3768
                               59  D4 00014             CLRL    R9                                              ;
                02       0D    A4  91 00016             CMPB    13(R4), #2                                      ;
                               2A  12 0001A             BNEQ    4$                                              ;
                               59  D6 0001C             INCL    R9                                              ;
                0C       90    AB  E9 0001E             BLBC    QUAL+14, 1$                                     ; 3772
                50       D1    AB  9A 00022             MOVZBL  QUAL+79, R0                                     ; 3773
              26 A4              50  B1 00026             CMPW    R0, 38(R4)                                      ;
                               0D  12 0002A             BNEQ    3$                                              ;
                               06  11 0002C             BRB     2$                                              ;
                01       26    A4  B1 0002E 1$:         CMPW    38(R4), #1                                      ; 3774
                               05  1A 00032             BGTRU   3$                                              ;
            F50C CF              00  FB 00034 2$:         CALLS   #0, BACKUP_SUMMARY                              ; 3776
                56             24  C0 00039 3$:         ADDL2   #36, L1                                         ; 3778
                01             6B  91 0003C             CMPB    COM_I_SETCOUNT, #1                              ; 3779
                               0A  1B 0003F             BLEQU   5$                                              ;
                56             06  C0 00041             ADDL2   #6, L1                                          ;
                               05  11 00044             BRB     5$                                              ; 3768
            F4FA CF              00  FB 00046 4$:         CALLS   #0, BACKUP_SUMMARY                              ; 3782
                58       08    AC  D0 0004B 5$:         MOVL    INPUT_BOOT, R8                                  ; 3788
                               5A  D4 0004F             CLRL    R10                                             ;
                         05    A8  95 00051             TSTB    5(R8)                                           ;
                               1C  12 00054             BNEQ    9$                                              ;
                               5A  D6 00056             INCL    R10                                             ;
                               52  D4 00058             CLRL    BOOT_LENGTH                                     ; 3791
                50       01FF  8F  3C 0005A             MOVZWL  #511, J                                         ; 3792
                             6048  95 0005F 6$:         TSTB    (J)[R8]                                         ; 3794
```

```
                            06  13 00062          BEQL    7$                                          ;
           52         01    A0  9E 00064          MOVAB   1(R0), BOOT_LENGTH                          ; 3797
                            03  11 00068          BRB     8$                                          ; 3796
           F2               50  F4 0006A  7$:     SOBGEQ  J, 6$                                       ; 3792
           56         04  A246  9E 0006D  8$:     MOVAB   4(BOOT_LENGTH)[L1], L1                      ; 3801
           50         0F    A6  9E 00072  9$:     MOVAB   15(R6), R0                                  ; 3807
  57       50               0F  CB 00076          BICL3   #15, R0, L2                                 ;
                            57  DD 0007A          PUSHL   L2                                          ; 3808
      F19C CF               01  FB 0007C          CALLS   #1, MAKE_SPACE                              ;
           53               50  D0 00081          MOVL    R0, P                                       ;
  83       57               10  A3 00084          SUBW3   #16, L2, (P)+                               ; 3809
           83               02  B0 00088          MOVW    #2, (P)+                                    ; 3810
                            83  7C 0008B          CLRQ    (P)+                                        ; 3811
                            83  D4 0008D          CLRL    (P)+                                        ; 3814
           83       0101    8F  B0 0008F          MOVW    #257, (P)+                                  ; 3816
           55         0C    AC  D0 00094          MOVL    INPUT_SCB, R5                               ; 3842
           03               59  E8 00098          BLBS    R9, 10$                                     ; 3845
                          011C  31 0009B          BRW     12$                                         ;
           83 00140002      8F  D0 0009E  10$:    MOVL    #1310722, (P)+                              ; 3824
           83         0C    A4  B0 000A5          MOVW    12(R4), (P)+                                ;
           83 0015000C      8F  D0 000A9          MOVL    #1376268, (P)+                              ; 3825
           83       01D8    C4  7D 000B0          MOVQ    472(R4), (P)+                               ;
           83       01E0    C4  D0 000B5          MOVL    480(R4), (P)+                               ;
           83 0016000C      8F  D0 000BA          MOVL    #1441804, (P)+                              ; 3826
           83       01E4    C4  7D 000C1          MOVQ    484(R4), (P)+                               ;
           83       01EC    C4  D0 000C6          MOVL    492(R4), (P)+                               ;
           83 0017000C      8F  D0 000CB          MOVL    #1507340, (P)+                              ; 3827
           83       01F0    C4  7D 000D2          MOVQ    496(R4), (P)+                               ;
           83       01F8    C4  D0 000D7          MOVL    504(R4), (P)+                               ;
           01               6B  91 000DC          CMPB    COM_I_SETCOUNT, #1                          ; 3828
                            0B  1B 000DF          BLEQU   11$                                         ;
           83 00180002      8F  D0 000E1          MOVL    #1572866, (P)+                              ; 3830
           83         26    A4  B0 000E8          MOVW    38(R4), (P)+                                ;
           83 00190004      8F  D0 000EC  11$:    MOVL    #1638404, (P)+                              ; 3831
           83         2C    A4  D0 000F3          MOVL    44(R4), (P)+                                ;
           83 001A0002      8F  D0 000F7          MOVL    #1703938, (P)+                              ; 3832
           83         34    A4  B0 000FE          MOVW    52(R4), (P)+                                ;
           83 001B0002      8F  D0 00102          MOVL    #1769474, (P)+                              ; 3833
           83         36    A4  B0 00109          MOVW    54(R4), (P)+                                ;
           83 001C0002      8F  D0 0010D          MOVL    #1835010, (P)+                              ; 3834
           83         38    A4  B0 00114          MOVW    56(R4), (P)+                                ;
           83 001D0002      8F  D0 00118          MOVL    #1900546, (P)+                              ; 3835
           83         2A    A4  B0 0011F          MOVW    42(R4), (P)+                                ;
           83 001E0008      8F  D0 00123          MOVL    #1966088, (P)+                              ; 3836
           83         3C    A4  7D 0012A          MOVQ    60(R4), (P)+                                ;
           83 001F0001      8F  D0 0012E          MOVL    #2031617, (P)+                              ; 3837
           83         44    A4  90 00135          MOVB    68(R4), (P)+                                ;
           83 00200001      8F  D0 00139          MOVL    #2097153, (P)+                              ; 3838
           83         45    A4  90 00140          MOVB    69(R4), (P)+                                ;
           83 00210002      8F  D0 00144          MOVL    #2162690, (P)+                              ; 3839
           83         46    A4  B0 0014B          MOVW    70(R4), (P)+                                ;
           83 00220002      8F  D0 0014F          MOVL    #2228226, (P)+                              ; 3840
           83         0E    A4  B0 00156          MOVW    14(R4), (P)+                                ;
           83 00230002      8F  D0 0015A          MOVL    #2293762, (P)+                              ; 3841
           83         22    A4  B0 00161          MOVW    34(R4), (P)+                                ;
           83 00240004      8F  D0 00165          MOVL    #2359300, (P)+                              ; 3842
           83         04    A5  D0 0016C          MOVL    4(R5), (P)+                                 ;
```

```
                        83 00270004   8F   D0 00170          MOVL    #2555908, (P)+                          ; 3843
                        83       1C   A4   D0 00177          MOVL    28(R4), (P)+
                        83 00280004   8F   D0 0017B          MOVL    #2621444, (P)+                          ; 3844
                        83       10   AC   D0 00182          MOVL    INPUT_MAXFILNUM, (P)+
                        83 00290004   8F   D0 00186          MOVL    #2686980, (P)+                          ; 3845
                        83     01C8   C4   D0 0018D          MOVL    456(R4), (P)+
                        83 00430004   8F   D0 00192          MOVL    #4390916, (P)+                          ; 3846
                        83       18   A4   D0 00199          MOVL    24(R4), (P)+
                        83 004C0008   8F   D0 0019D          MOVL    #4980744, (P)+                          ; 3847
                        83       48   A4   7D 001A4          MOVQ    72(R4), (P)+
                        83 004D0008   8F   D0 001A8          MOVL    #5046280, (P)+                          ; 3848
                        83       50   A4   D0 001AF          MOVL    80(R4), (P)+
                        63       54   A4   D0 001B3          MOVL    84(R4), (P)
                                 00F5   31   001B7          BRW     13$                                     ; 3821
                        83 00140002   8F   D0 001BA 12$:     MOVL    #1310722, (P)+                          ; 3856
                        83       0C   A4   B0 001C1          MOVW    12(R4), (P)+
                        83 0015000C   8F   D0 001C5          MOVL    #1376268, (P)+                          ; 3857
                        83     01D8   C4   7D 001CC          MOVQ    472(R4), (P)+
                        83     01E0   C4   D0 001D1          MOVL    480(R4), (P)+
                        83 0016000C   8F   D0 001D6          MOVL    #1441804, (P)+                          ; 3858
                        83     01E4   C4   7D 001DD          MOVQ    484(R4), (P)+
                        83     01EC   C4   D0 001E2          MOVL    492(R4), (P)+
                        83 0017000C   8F   D0 001E7          MOVL    #1507340, (P)+                          ; 3859
                        83     01F0   C4   7D 001EE          MOVQ    496(R4), (P)+
                        83     01F8   C4   D0 001F3          MOVL    504(R4), (P)+
                        50       1E   A4   9B 001F8          MOVZBW  30(R4), T                               ; 3860
                        51       1F   A4   9A 001FC          MOVZBL  31(R4), R1                              ; 3861
              50                10        51   F0 00200          INSV    R1, #16, #16, T
                        83 00190004   8F   D0 00205          MOVL    #1638404, (P)+                          ; 3862
                        83       50   D0 0020C          MOVL    T, (P)+
                        83 001A0002   8F   D0 0020F          MOVL    #1703938, (P)+                          ; 3863
                        83       20   A4   B0 00216          MOVW    32(R4), (P)+
                        83 001B0002   8F   D0 0021A          MOVL    #1769474, (P)+                          ; 3864
                        83       24   A4   B0 00221          MOVW    36(R4), (P)+
                        83 001D0002   8F   D0 00225          MOVL    #1900546, (P)+                          ; 3865
                        83       2A   A4   B0 0022C          MOVW    42(R4), (P)+
                                      5E   DD 00230          PUSHL   SP                                      ; 3866
                                 3C   A4   9F 00232          PUSHAB  60(R4)
                           F489  CF         02   FB 00235          CALLS   #2, FROM_ODS1_DATE
                        83 001E0008   8F   D0 0023A          MOVL    #1966088, (P)+                          ; 3867
                        83            6E   7D 00241          MOVQ    CREDATE, (P)+
                        83 001F0001   8F   D0 00244          MOVL    #2031617, (P)+                          ; 3868
                        83       2C   A4   90 0024B          MOVB    44(R4), (P)+
                        83 00200001   8F   D0 0024F          MOVL    #2097153, (P)+                          ; 3869
                        83       2E   A4   90 00256          MOVB    46(R4), (P)+
                        83 00210002   8F   D0 0025A          MOVL    #2162690, (P)+                          ; 3870
                        83       2D   A4   9B 00261          MOVZBW  45(R4), (P)+
                        83 00220002   8F   D0 00265          MOVL    #2228226, (P)+                          ; 3871
                        83       08   A4   B0 0026C          MOVW    8(R4), (P)+
                        83 00240004   8F   D0 00270          MOVL    #2359300, (P)+                          ; 3872
                        50       03   A5   9A 00277          MOVZBL  3(R5), R0
              83     04 A540               10   9C 0027B          ROTL    #16, 4(R5)[R0], (P)+
                        83 00270004   8F   D0 00281          MOVL    #2555908, (P)+                          ; 3873
                        83       06   A4   3C 00288          MOVZWL  6(R4), (P)+
                        83 00280004   8F   D0 0028C          MOVL    #2621444, (P)+                          ; 3874
                        83       10   AC   D0 00293          MOVL    INPUT_MAXFILNUM, (P)+
                        83 00290004   8F   D0 00297          MOVL    #2686980, (P)+                          ; 3875
```

```
                                83      01C8    C4  D0 0029E          MOVL    456(R4), (P)+                                ;
                                83 00430004     8F  D0 002A3          MOVL    #4390916, (P)+                               ; 3876
                    63      02  A4              10  9C 002AA          ROTL    #16, 2(R4), (P)                              ;
                                53              04  C0 002AF 13$:     ADDL2   #4, P                                        ; 3848
                                0B              5A  E9 002B2          BLBC    R10, 14$                                     ; 3880
                                83              52  B0 002B5          MOVW    BOOT_LENGTH, (P)+                            ; 3882
                                83      44      8F  9B 002B8          MOVZBW  #68, (P)+                                    ;
                    63          68              52  28 002BC          MOVC3   BOOT_LENGTH, (R8), (P)                       ;
                                57              56  C2 002C0 14$:     SUBL2   L1, R7                                       ; 3887
            57      00          6E              00  2C 002C3          MOVC5   #0, (SP), #0, R7, (P)                        ;
                                                63      002C8                                                            ;
                                                    04 002C9          RET                                                  ; 3888

; Routine Size:  714 bytes,    Routine Base:  CODE + 0F83
```

```
;  2355          3889  1 %SBTTL 'GEN_FID_RECORD - format file ID record'
;  2356          3890  1 GLOBAL ROUTINE GEN_FID_RECORD(BUFFER,COUNT,FILE_NUMBER,RVN): NOVALUE=
;  2357          3891  1
;  2358          3892  1 !++
;  2359          3893  1 !
;  2360          3894  1 ! FUNCTIONAL DESCRIPTION:
;  2361          3895  1 !       This routine generates a file ID record.
;  2362          3896  1 !
;  2363          3897  1 ! INPUT PARAMETERS:
;  2364          3898  1 !       BUFFER          - Buffer containing file headers.
;  2365          3899  1 !       COUNT           - Count of file headers in buffer.
;  2366          3900  1 !       FILE_NUMBER     - File number and RVN of first
;  2367          3901  1 !       RVN             -     header in buffer.
;  2368          3902  1 !
;  2369          3903  1 ! IMPLICIT INPUTS:
;  2370          3904  1 !       NONE
;  2371          3905  1 !
;  2372          3906  1 ! OUTPUT PARAMETERS:
;  2373          3907  1 !       NONE
;  2374          3908  1 !
;  2375          3909  1 ! IMPLICIT OUTPUTS:
;  2376          3910  1 !       NONE
;  2377          3911  1 !
;  2378          3912  1 ! ROUTINE VALUE:
;  2379          3913  1 !       NONE
;  2380          3914  1 !
;  2381          3915  1 ! SIDE EFFECTS:
;  2382          3916  1 !       File ID record generated.
;  2383          3917  1 !
;  2384          3918  1 !--
;  2385          3919  1
;  2386          3920  2 BEGIN
;  2387          3921  2 LOCAL
;  2388          3922  2         L1,                             ! Unrounded length
;  2389          3923  2         L2,                             ! Rounded length
;  2390          3924  2         P:              REF BBLOCK,     ! Pointer to attribute record
;  2391          3925  2         HDR:            REF BBLOCK;     ! Pointer to file header
;  2392          3926  2
;  2393          3927  2
;  2394          3928  2 ! Point to the file headers.
;  2395          3929  2 !
;  2396          3930  2 HDR = .BUFFER;
;  2397          3931  2
;  2398          3932  2
;  2399          3933  2 ! Determine the length of the attribute record.
;  2400          3934  2 !
;  2401          3935  2 L1 = BRH$C_LENGTH + $BYTEOFFSET(BSA$W_FID_SEQ) + 2 * .COUNT;
;  2402          3936  2 L2 = (.L1 + 15) AND NOT 15;
;  2403          3937  2
;  2404          3938  2
;  2405          3939  2 ! Allocate the attribute record and initialize the record header.
;  2406          3940  2 !
;  2407          3941  2 P = MAKE_SPACE(.L2);
;  2408          3942  2 P[BRH$W_RSIZE] = .L2 - BRH$C_LENGTH;
;  2409          3943  2 P[BRH$W_RTYPE] = BRH$K_FID;
;  2410          3944  2 P[BRH$L_FLAGS] = 0;
;  2411          3945  2 P[BRH$L_ADDRESS] = 0;
```

```
;  2412     3946  2 $ASSUME ($BYTEOFFSET (BRH$W_BLOCKFLAGS) + 4, EQL, BRH$K_LENGTH);
;  2413     3947  2 (P[BRH$W_BLOCKFLAGS])<0,32> = 0;
;  2414     3948  2 P = .P + BRH$C_LENGTH;
;  2415     3949  2 STORE_(2, BBH$K_LEVEL1);
;  2416     3950  2 STORE_(2, .FILE_NUMBER<0,16>);
;  2417     3951  2 STORE_(1, .RVN);
;  2418     3952  2 STORE_(1, .FILE_NUMBER<16,8>);
;  2419     3953  2 STORE_(2, .COUNT);
;  2420     3954  2 DECR I FROM .COUNT TO 1 DO
;  2421     3955  3     BEGIN
;  2422     3956  3     LOCAL
;  2423     3957  3         SEQ;
;  2424     3958  3
;  2425     3959  3     IF .HDR[FH2$B_STRUCLEV] EQL 2
;  2426     3960  3         THEN SEQ = .HDR[FH2$W_FID_SEQ]
;  2427     3961  3         ELSE SEQ = .HDR[FH1$W_FID_SEQ];
;  2428     3962  3     STORE_(2, .SEQ);
;  2429     3963  3     HDR = .HDR + 512;
;  2430     3964  2     END;
;  2431     3965  2
;  2432     3966  2
;  2433     3967  2 ! Pad the record with zeros.
;  2434     3968  2 !
;  2435     3969  2 CH$FILL(0, .L2 - .L1, .P);
;  2436     3970  1 END;
```

```
                                    003C 00000          .ENTRY  GEN_FID_RECORD, Save R2,R3,R4,R5          ; 3890
                          55    04  AC  D0 00002          MOVL    BUFFER, HDR                               ; 3930
                          54    08  AC  D0 00006          MOVL    COUNT, R4                                 ; 3935
                    53    54        01  78 0000A          ASHL    #1, R4, L1                                ;
                          53        18  C0 0000E          ADDL2   #24, L1                                   ;
                          51    0F  A3  9E 00011          MOVAB   15(R3), R1                                ; 3936
                    52    51        0F  CB 00015          BICL3   #15, R1, L2                               ;
                                    52  DD 00019          PUSHL   L2                                        ; 3941
                EF33      CF        01  FB 0001B          CALLS   #1, MAKE_SPACE                            ;
                    80    52        10  A3 00020          SUBW3   #16, L2, (P)+                             ; 3942
                          80        07  B0 00024          MOVW    #7, (P)+                                  ; 3943
                                    80  7C 00027          CLRQ    (P)+                                      ; 3944
                                    80  D4 00029          CLRL    (P)+                                      ; 3947
                          80  0101  8F  B0 0002B          MOVW    #257, (P)+                                ; 3949
                          80    0C  AC  B0 00030          MOVW    FILE_NUMBER, (P)+                         ; 3950
                          80    10  AC  90 00034          MOVB    RVN, (P)+                                 ; 3951
                          80    0E  AC  90 00038          MOVB    FILE_NUMBER+2, (P)+                       ; 3952
                          80        54  B0 0003C          MOVW    R4, (P)+                                  ; 3953
                          51    01  A4  9E 0003F          MOVAB   1(R4), I                                  ; 3954
                                    18  11 00043          BRB     4$                                        ;
                          02    07  A5  91 00045 1$:      CMPB    7(HDR), #2                                ; 3959
                                    06  12 00049          BNEQ    2$                                        ;
                          54    0A  A5  3C 0004B          MOVZWL  10(HDR), SEQ                              ; 3960
                                    04  11 0004F          BRB     3$                                        ;
                          54    04  A5  3C 00051 2$:      MOVZWL  4(HDR), SEQ                               ; 3961
                          80        54  B0 00055 3$:      MOVW    SEQ, (P)+                                 ; 3962
                          55  0200  C5  9E 00058          MOVAB   512(R5), HDR                              ; 3963
```

```
                         E5          51  F5 0005D 4$:    SOBGTR  I, 1$                          ; 3954
                         52          53  C2 00060        SUBL2   L1, R2                        ; 3969
          52       00    6E          00  2C 00063        MOVC5   #0, (SP), #0, R2, (P)         ;
                                       60     00068                                           ;
                                          04 00069        RET                                   ; 3970

; Routine Size:  106 bytes,    Routine Base:  CODE + 124D
```

```
;  2438     3971  1 %SBTTL 'PHYSVOL_SUMMARY - format physical volume summary record'
;  2439     3972  1 ROUTINE PHYSVOL_SUMMARY(INPUT_DEVCHAR): NOVALUE=
;  2440     3973  1
;  2441     3974  1 !++
;  2442     3975  1 !
;  2443     3976  1 ! FUNCTIONAL DESCRIPTION:
;  2444     3977  1 !       This routine generates the physical volume summary record.
;  2445     3978  1 !
;  2446     3979  1 ! INPUT PARAMETERS.
;  2447     3980  1 !       INPUT_DEVCHAR   - Device characteristics.
;  2448     3981  1 !
;  2449     3982  1 ! IMPLICIT INPUTS:
;  2450     3983  1 !       NONE
;  2451     3984  1 !
;  2452     3985  1 ! OUTPUT PARAMETERS:
;  2453     3986  1 !       NONE
;  2454     3987  1 !
;  2455     3988  1 ! IMPLICIT OUTPUTS:
;  2456     3989  1 !       NONE
;  2457     3990  1 !
;  2458     3991  1 ! ROUTINE VALUE:
;  2459     3992  1 !       NONE
;  2460     3993  1 !
;  2461     3994  1 ! SIDE EFFECTS:
;  2462     3995  1 !       NONE
;  2463     3996  1 !
;  2464     3997  1 !--
;  2465     3998  1
;  2466     3999  2 BEGIN
;  2467     4000  2 MAP
;  2468     4001  2         INPUT_DEVCHAR:  REF BBLOCK;     ! Pointer to device characteristics
;  2469     4002  2 LOCAL
;  2470     4003  2         L1,                             ! Unrounded length
;  2471     4004  2         L2,                             ! Rounded length
;  2472     4005  2         P;                              ! Pointer to attribute record
;  2473     4006  2 LITERAL
;  2474     4007  2     FIXED_SIZE =
;  2475     4008  2         BRH$C_LENGTH +          ! record header
;  2476     4009  2         2 +                     ! structure level
;  2477     4010  2         7*4 +                   ! 7 attribute size/type longwords
;  2478     4011  2         BSA$S_SECTORS +         !  1. sectors per track
;  2479     4012  2         BSA$S_TRACKS +          !  2. tracks per cylinder
;  2480     4013  2         BSA$S_CYLINDERS +       !  3. cylinders per volume
;  2481     4014  2         BSA$S_MAXBLOCK +        !  4. logical blocks per volume
;  2482     4015  2         BSA$S_DEVTYP +          !  5. device type
;  2483     4016  2         BSA$S_SERIAL;           !  6. serial number
;  2484     4017  2                                 !  7. device name (variable)
;  2485     4018  2
;  2486     4019  2 $ASSUME (BSA$S_DEVTYP, EQL, 4) ;
;  2487     4020  2
;  2488     4021  2 L1 =
;  2489     4022  2     FIXED_SIZE +
;  2490     4023  2     .BBLOCK[INPUT_QUAL[QUAL_DVI_DESC], DSC$W_LENGTH] + 1;
;  2491     4024  2
;  2492     4025  2
;  2493     4026  2 IF .INPUT_DEVCHAR[DIB$W_VOLNAMOFF] NEQ 0
;  2494     4027  2 THEN
```

```
;2495        4028  2      L1 = .L1 + 4 + .(.INPUT_DEVCHAR + .INPUT_DEVCHAR[DIB$W_VOLNAMOFF])<0,8>;
;2496        4029  2
;2497        4030  2
;2498        4031  2  IF .INPUT_BAD[BAD_NUMDESC] NEQ 0
;2499        4032  2  THEN
;2500        4033  2      L1 = .L1 + 4 + BSA$S_BADBLOCK * .INPUT_BAD[BAD_NUMDESC];
;2501        4034  2
;2502        4035  2
;2503        4036  2  L2 = (.L1 + 15) AND NOT 15;
;2504        4037  2  P = MAKE_SPACE(.L2);
;2505        4038  2
;2506        4039  2
;2507        4040  2  STORE_(2, .L2 - BRH$C_LENGTH);
;2508        4041  2  STORE_(2, BRH$K_PHYSVOL);
;2509        4042  2  STORE_(4, 0);
;2510        4043  2  STORE_(4, 0);
;2511        4044  2  STORE_(4, 0);
;2512        4045  2  STORE_(2, BBH$K_LEVEL1);
;2513        4046  2
;2514        4047  2  ! Call GETDVI to get the MEDIA_ID from the UCB.  We save the longword
;2515        4048  2  ! encoded form of the media ID instead of the ascii name and type which
;2516        4049  2  ! can be obtained via DVI$_MEDIA_NAME and DVI$_MEDIA_TYPE.  This is done
;2517        4050  2  ! because the HSC folks are using this form for backup savesets created by
;2518        4051  2  ! the HSC.
;2519        4052  2  !
;2520      P 4053  2  $GETDVI(CHAN    = .INPUT_CHAN ,
;2521      P 4054  2          ITMLST = UPLIT(
;2522      P 4055  2                   WORD(4, DVI$_MEDIA_ID),
;2523      P 4056  2                   INPUT_MEDIA_ID,
;2524      P 4057  2                   0,
;2525        4058  2                   0)) ;
;2526        4059  2
;2527        4060  2  ATT_('SECTORS',   .INPUT_DEVCHAR[DIB$B_SECTORS]);
;2528        4061  2  ATT_('TRACKS',    .INPUT_DEVCHAR[DIB$B_TRACKS]);
;2529        4062  2  ATT_('CYLINDERS',.INPUT_DEVCHAR[DIB$W_CYLINDERS]);
;2530        4063  2  ATT_('MAXBLOCK', .INPUT_DEVCHAR[DIB$L_MAXBLOCK]);
;2531        4064  2  ATT_('DEVTYP', .INPUT_MEDIA_ID) ;
;2532        4065  2  ATT_('SERIAL',    .INPUT_BAD[BAD_SERIAL]);
;2533        4066  2
;2534        4067  2
;2535      P 4068  2  DVI_('DEVNAM',
;2536      P 4069  2      .BBLOCK[INPUT_QUAL[QUAL_DVI_DESC], DSC$W_LENGTH],
;2537        4070  2      .BBLOCK[INPUT_QUAL[QUAL_DVI_DESC], DSC$A_POINTER]);
;2538        4071  2
;2539        4072  2
;2540        4073  2  IF .INPUT_DEVCHAR[DIB$W_VOLNAMOFF] NEQ 0
;2541        4074  2  THEN
;2542      P 4075  2      ATV_('LABEL',
;2543      P 4076  2      .(.INPUT_DEVCHAR + .INPUT_DEVCHAR[DIB$W_VOLNAMOFF])<0,8>,
;2544        4077  2      .INPUT_DEVCHAR + .INPUT_DEVCHAR[DIB$W_VOLNAMOFF] + 1);
;2545        4078  2
;2546        4079  2
;2547        4080  2  IF .INPUT_BAD[BAD_NUMDESC] NEQ 0
;2548        4081  2  THEN
;2549        4082  3      BEGIN
;2550        4083  3      LOCAL
;2551        4084  3          Q:      REF BBLOCK;
```

```
;  2552          4085  3
;  2553          4086  3      STORE_(2, BSA$S_BADBLOCK * .INPUT_BAD[BAD_NUMDESC]);
;  2554          4087  3      STORE_(2, BSA$K_BADBLOCK);
;  2555          4088  3      Q = INPUT_BAD[BAD_DESC];
;  2556          4089  3      INCR I FROM 0 TO .INPUT_BAD[BAD_NUMDESC]-1 DO
;  2557          4090  4          BEGIN
;  2558          4091  4          STORE_(4, .Q[BAD_LBN]);
;  2559          4092  4          STORE_(4, .Q[BAD_COUNT]);
;  2560          4093  4          Q = .Q + BAD_S_DESC;
;  2561          4094  3          END;
;  2562          4095  2      END;
;  2563          4096  2
;  2564          4097  2
;  2565          4098  2 CH$FILL(0, .L2 - .L1, .P);
;  2566          4099  1 END;


                                                012B7          .BLKB   1
                                        011A  0004  012B8 P.AAC:  .WORD   4, 282                                          ;
                                          00000000' 012BC         .ADDRESS INPUT_MEDIA_ID                                 ;
                                 00000000 00000000  012C0         .LONG   0, 0                                            ;

                                                                  .EXTRN  SYS$GETDVI

                                                    0FFC 00000 PHYSVOL_SUMMARY:
                                                               .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11                ; 3972
                                    50 00000000'   EF   D0 00002         MOVL    INPUT_QUAL, R0                          ; 4023
                                    56       18    A0   3C 00009         MOVZWL  24(R0), L1                              ;
                                    56             3F   C0 0000D         ADDL2   #63, L1                                 ;
                                    58       04    AC   D0 00010         MOVL    INPUT_DEVCHAR, R8                       ; 4026
                                    5A       20    A8   3C 00014         MOVZWL  32(R8), R10                             ;
                                                   5B   D4 00018         CLRL    R11                                     ;
                                                   5A   D5 0001A         TSTL    R10                                     ;
                                                   0B   13 0001C         BEQL    1$                                      ;
                                                   5B   D6 0001E         INCL    R11                                     ;
                                          50             6A48 9A 00020         MOVZBL  (R10)[R8], R0                       ; 4028
                                          56       04    A046 9E 00024         MOVAB   4(R0)[L1], L1                       ;
                                          50 00000000'   FF   D0 00029 1$:      MOVL    @INPUT_BAD, R0                      ; 4031
                                                   05   13 00030         BEQL    2$                                      ;
                                          56       04    A640 7E 00032         MOVAQ   4(L1)[R0], L1                       ; 4033
                                          50       0F    A6   9E 00037 2$:     MOVAB   15(R6), R0                          ; 4036
                                57        50             0F   CB 0003B         BICL3   #15, R0, L2                         ;
                                                   57   DD 0003F         PUSHL   L2                                      ; 4037
                                      EE92 CF             01   FB 00041         CALLS   #1, MAKE_SPACE                      ;
                                          52             50   D0 00046         MOVL    R0, P                               ;
                                82        57             10   A3 00049         SUBW3   #16, L2, (P)+                       ; 4040
                                82                       05   B0 0004D         MOVW    #5, (P)+                            ; 4041
                                                         82   7C 00050         CLRQ    (P)+                                ; 4042
                                                         82   D4 00052         CLRL    (P)+                                ; 4044
                                82             0101      8F   B0 00054         MOVW    #257, (P)+                          ; 4045
                                                         7E   7C 00059         CLRQ    -(SP)                               ; 4058
                                                         7E   7C 0005B         CLRQ    -(SP)                               ;
                                                   90    AF   9F 0005D         PUSHAB  P.AAC                               ;
                                                         7E   D4 00060         CLRL    -(SP)                               ;
                                             00000000'   EF   DD 00062         PUSHL   INPUT_CHAN                          ;
                                                         7E   D4 00068         CLRL    -(SP)                               ;
```

```
                 00000000G  00              08  FB 0006A           CALLS   #8, SYS$GETDVI                                   ;
                            82 003A0001      8F  D0 00071           MOVL    #3801089, (P)+                                  ; 4060
                            82       08      A8  90 00078           MOVB    8(R8), (P)+                                     ;
                            82 003B0001      8F  D0 0007C           MOVL    #3866625, (P)+                                  ; 4061
                            82       09      A8  90 00083           MOVB    9(R8), (P)+                                     ;
                            82 003C0002      8F  D0 00087           MOVL    #3932162, (P)+                                  ; 4062
                            82       0A      A8  B0 0008E           MOVW    10(R8), (P)+                                    ;
                            82 003D0004      8F  D0 00092           MOVL    #3997700, (P)+                                  ; 4063
                            82       70      A8  D0 00099           MOVL    112(R8), (P)+                                   ;
                            82 003E0004      8F  D0 0009D           MOVL    #4063236, (P)+                                  ; 4064
                            82 00000000'     EF  D0 000A4           MOVL    INPUT_MEDIA_ID, (P)+                            ;
                            82 003F0004      8F  D0 000AB           MOVL    #4128772, (P)+                                  ; 4065
                            59 00000000'     EF  D0 000B2           MOVL    INPUT_BAD, R9                                   ;
                            82       04      A9  D0 000B9           MOVL    4(R9), (P)+                                     ;
                            50 00000000'     EF  D0 000BD           MOVL    INPUT_QUAL, R0                                  ; 4070
                            50               18  C0 000C4           ADDL2   #24, R0                                         ;
                         82                  01  A1 000C7           ADDW3   #1, (R0), (P)+                                  ;
                            82       40      8F  9B 000CB           MOVZBW  #64, (P)+                                       ;
                         62       04 B0      60  28 000CF           MOVC3   (R0), @4(R0), (P)                               ;
                            52               53  D0 000D4           MOVL    R3, P                                           ;
                            82               3A  90 000D7           MOVB    #58, (P)+                                       ;
                            16               5B  E9 000DA           BLBC    R11, 3$                                         ; 4073
                         50 58               5A  C1 000DD           ADDL3   R10, R8, R0                                     ; 4077
                            82               60  9B 000E1           MOVZBW  (R0), (P)+                                      ;
                            82       41      8F  9B 000E4           MOVZBW  #65, (P)+                                       ;
                            51               60  9A 000E8           MOVZBL  (R0), R1                                        ;
                         62       01 A0      51  28 000EB           MOVC3   R1, 1(R0), (P)                                  ;
                            52               53  D0 000F0           MOVL    R3, P                                           ;
                                             69  D5 000F3 3$:       TSTL    (R9)                                            ; 4080
                                             18  13 000F5           BEQL    6$                                              ;
                         82                  69  08 A5 000F7         MULW3   #8, (R9), (P)+                              ; 4086
                            82       42      8F  9B 000FB           MOVZBW  #66, (P)+                                       ; 4087
                            50       08      A9  9E 000FF           MOVAB   8(R9), Q                                        ; 4088
                            51               01  CE 00103           MNEGL   #1, I                                           ; 4089
                                             03  11 00106           BRB     5$                                              ;
                            82               80  7D 00108 4$:       MOVQ    (Q)+, (P)+                                      ; 4091
                         F9 51               69  F2 0010B 5$:       AOBLSS  (R9), I, 4$                                     ; 4089
                            57               56  C2 0010F 6$:       SUBL2   L1, R7                                          ; 4098
                  57    00 6E               00  2C 00112           MOVC5   #0, (SP), #0, R7, (P)                           ;
                                             62      00117                                                                   ;
                                                 04 00118           RET                                                     ; 4099

; Routine Size:  281 bytes,    Routine Base:  CODE + 12C8
```

```
;2568          4100  1 %SBTTL 'SAVE_VERIFY_REEL - verify save set volume'
;2569          4101  1 GLOBAL ROUTINE SAVE_VERIFY_REEL: NOVALUE=
;2570          4102  1
;2571          4103  1 !++
;2572          4104  1 !
;2573          4105  1 ! FUNCTIONAL DESCRIPTION:
;2574          4106  1 !       This routine is called at the end of each reel to verify that reel.
;2575          4107  1 !
;2576          4108  1 ! INPUT PARAMETERS:
;2577          4109  1 !       NONE
;2578          4110  1 !
;2579          4111  1 ! IMPLICIT INPUTS:
;2580          4112  1 !       NONE
;2581          4113  1 !
;2582          4114  1 ! OUTPUT PARAMETERS:
;2583          4115  1 !       NONE
;2584          4116  1 !
;2585          4117  1 ! IMPLICIT OUTPUTS:
;2586          4118  1 !       NONE
;2587          4119  1 !
;2588          4120  1 ! ROUTINE VALUE:
;2589          4121  1 !       NONE
;2590          4122  1 !
;2591          4123  1 ! SIDE EFFECTS:
;2592          4124  1 !       NONE
;2593          4125  1 !
;2594          4126  1 !--
;2595          4127  1
;2596          4128  2 BEGIN
;2597          4129  2 LOCAL
;2598          4130  2         STATUS,                         ! Status variable
;2599          4131  2         IOSB:           VECTOR[4,WORD], ! I/O status block
;2600          4132  2         SAVE_INPUT_BAD,                 ! Save for INPUT_BAD
;2601          4133  2         LOCAL_SAVE:     BBLOCK[INPUT_END-INPUT_BEG];    ! Save area
;2602          4134  2
;2603          4135  2
;2604          4136  2 ! If a file is accessed, deaccess it for the duration of the verify pass to
;2605          4137  2 ! avoid having the file open for the entire time.
;2606          4138  2 !
;2607          4139  2 IF .INPUT_FLAGS[INPUT_OPEN]
;2608          4140  2 THEN
;2609          4141  3     BEGIN
;2610        P 4142  3     STATUS = $QIOW(
;2611        P 4143  3         FUNC=IO$_DEACCESS,
;2612        P 4144  3         CHAN=.INPUT_CHAN,
;2613          4145  3         IOSB=IOSB);
;2614          4146  3     IF .STATUS THEN STATUS = .IOSB[0];
;2615          4147  3     IF NOT .STATUS
;2616          4148  3     THEN
;2617          4149  3         FILE_ERROR(
;2618          4150  3             BACKUP$_CLOSEIN + STS$K_ERROR,
;2619          4151  3             .INPUT_FAB,
;2620          4152  3             .STATUS);
;2621          4153  2     END;
;2622          4154  2
;2623          4155  2
;2624          4156  2 ! Save current context.
```

```
;  2625          4157  2 !
;  2626          4158  2 CH$MOVE(INPUT_END-INPUT_BEG, INPUT_BEG, LOCAL_SAVE);
;  2627          4159  2
;  2628          4160  2
;  2629          4161  2 ! Deassign the channel.  This is necessary so that if a restart is requested
;  2630          4162  2 ! during the verify pass, the channel will not be lost.
;  2631          4163  2 !
;  2632          4164  2 IF .INPUT_CHAN NEQ 0
;  2633          4165  2 THEN
;  2634          4166  3     BEGIN
;  2635          4167  3     $DASSGN(CHAN=.INPUT_CHAN);
;  2636          4168  3     .INPUT_CHAN = 0;
;  2637          4169  2     END;
;  2638          4170  2
;  2639          4171  2
;  2640          4172  2 ! Readjust context.
;  2641          4173  2 !
;  2642          4174  2 QUAL[QUAL_COMP] = TRUE;
;  2643          4175  2 IF .QUAL[QUAL_PHYS] THEN VERIFY_FAB = .INPUT_FAB;
;  2644          4176  2 SAVE_INPUT_BAD = .INPUT_BAD;
;  2645          4177  2 CH$FILL(0, INPUT_END-INPUT_BEG, INPUT_BEG);
;  2646          4178  2 CH$FILL(0, OUTPUT_END-OUTPUT_BEG, OUTPUT_BEG);
;  2647          4179  2 OUTPUT_BAD = .SAVE_INPUT_BAD;
;  2648          4180  2
;  2649          4181  2
;  2650          4182  2 ! Do the verify by calling the RESTORE module.
;  2651          4183  2 !
;  2652          4184  2 RESTORE();
;  2653          4185  2
;  2654          4186  2
;  2655          4187  2 ! Restore context.
;  2656          4188  2 !
;  2657          4189  2 QUAL[QUAL_COMP] = FALSE;
;  2658          4190  2 CH$MOVE(INPUT_END-INPUT_BEG, LOCAL_SAVE, INPUT_BEG);
;  2659          4191  2 OUTPUT_BAD = 0;
;  2660          4192  2
;  2661          4193  2
;  2662          4194  2 ! Reassign a channel to the input device if one was originally assigned.
;  2663          4195  2 !
;  2664          4196  2 IF .INPUT_CHAN NEQ 0
;  2665          4197  2 THEN
;  2666          4198  3     BEGIN
;  2667          4199  3     STATUS = ASSIGN_INPUT_CHANNEL(INPUT_QUAL[QUAL_DEV_DESC], INPUT_CHAN, 0, 0);
;  2668          4200  3     IF NOT .STATUS
;  2669          4201  3     THEN
;  2670          4202  3         FILE_ERROR(
;  2671          4203  3             BACKUP$_OPENIN + STS$K_SEVERE,
;  2672          4204  3             .INPUT_FAB,
;  2673          4205  3             .STATUS);
;  2674          4206  2     END;
;  2675          4207  1 END;

                                                        .EXTRN  SYS$DASSGN

                                03FC 00000              .ENTRY  SAVE_VERIFY_REEL, Save R2,R3,R4,R5,R6,R7,-  ; 4101
```

```
                                                            R8,R9                             ;
                59 00000000G  00  9E 00002        MOVAB   FILE_ERROR, R9                    ;
                58 00000000'  EF  9E 00009        MOVAB   INPUT_CHAN, R8                    ;
                5E      FF78  CE  9E 00010        MOVAB   -136(SP), SP                      ;
                33        04  A8  E9 00015        BLBC    INPUT_FLAGS, 2$                   ; 4139
                              7E  7C 00019        CLRQ    -(SP)                             ; 4145
                              7E  7C 0001B        CLRQ    -(SP)                             ;
                              7E  7C 0001D        CLRQ    -(SP)                             ;
                              7E  7C 0001F        CLRQ    -(SP)                             ;
                          F8  AD  9F 00021        PUSHAB  IOSB                              ;
                              34  DD 00024        PUSHL   #52                               ;
                              68  DD 00026        PUSHL   INPUT_CHAN                        ;
                              7E  D4 00028        CLRL    -(SP)                             ;
            00000000G  00     0C  FB 0002A        CALLS   #12, SYS$QIOW                     ;
                       57     50  D0 00031        MOVL    R0, STATUS                        ;
                       07     57  E9 00034        BLBC    STATUS, 1$                        ; 4146
                       57  F8 AD  3C 00037        MOVZWL  IOSB, STATUS                      ;
                       0E     57  E8 0003B        BLBS    STATUS, 2$                        ; 4147
                              57  DD 0003E 1$:    PUSHL   STATUS                            ; 4152
                          08  A8  DD 00040        PUSHL   INPUT_FAB                         ; 4151
                   00000000G  8F  DD 00043        PUSHL   #BACKUP$_CLOSEIN+2                ; 4150
                       69     03  FB 00049        CALLS   #3, FILE_ERROR                    ;
         6E            68   0080  8F  28 0004C 2$: MOVC3  #128, INPUT_BEG, LOCAL_SAVE       ; 4158
                       50     68  D0 00052        MOVL    INPUT_CHAN, R0                    ; 4164
                              0B  13 00055        BEQL    3$                                ;
                              50  DD 00057        PUSHL   R0                                ; 4167
            00000000G  00     01  FB 00059        CALLS   #1, SYS$DASSGN                    ;
                              68  D4 00060        CLRL    INPUT_CHAN                        ; 4168
               FF44    C8     80  8F  88 00062 3$: BISB2  #128, QUAL+8                      ; 4174
         06    FF48    C8         05  E1 00068    BBC     #5, QUAL+12, 4$                   ; 4175
               015C    C8     08  A8  D0 0006E    MOVL    INPUT_FAB, VERIFY_FAB             ;
                       56     18  A8  D0 00074 4$: MOVL   INPUT_BAD, SAVE_INPUT_BAD         ; 4176
0080   8F      00             6E  00  2C 00078    MOVC5   #0, (SP), #0, #128, INPUT_BEG     ; 4177
                                  68     0007F                                              ;
00BC   8F      00             6E  00  2C 00080    MOVC5   #0, (SP), #0, #188, OUTPUT_BEG    ; 4178
                          0098    C8     00087                                              ;
               00B0    C8         56  D0 0008A    MOVL    SAVE_INPUT_BAD, OUTPUT_BAD        ; 4179
            00000000G  00         00  FB 0008F    CALLS   #0, RESTORE                       ; 4184
               FF44    C8     80  8F  8A 00096    BICB2   #128, QUAL+8                      ; 4189
         68            6E   0080  8F  28 0009C    MOVC3   #128, LOCAL_SAVE, INPUT_BEG       ; 4190
                            00B0  C8  D4 000A2    CLRL    OUTPUT_BAD                        ; 4191
                                  68  D5 000A6    TSTL    INPUT_CHAN                        ; 4196
                                  24  13 000A8    BEQL    5$                                ;
                                  7E  7C 000AA    CLRQ    -(SP)                             ; 4199
                                  58  DD 000AC    PUSHL   R8                                ;
         7E       14   A8         10  C1 000AE    ADDL3   #16, INPUT_QUAL, -(SP)            ;
            00000000G  00         04  FB 000B3    CALLS   #4, ASSIGN_INPUT_CHANNEL          ;
                       57         50  D0 000BA    MOVL    R0, STATUS                        ;
                       0E         57  E8 000BD    BLBS    STATUS, 5$                        ; 4200
                                  57  DD 000C0    PUSHL   STATUS                            ; 4205
                              08  A8  DD 000C2    PUSHL   INPUT_FAB                         ; 4204
                       00000000G  8F  DD 000C5    PUSHL   #BACKUP$_OPENIN+4                 ; 4203
                       69         03  FB 000CB    CALLS   #3, FILE_ERROR                    ;
                                  04     000CE 5$: RET                                      ; 4207

; Routine Size:  207 bytes,    Routine Base:  CODE + 13E1
```

```
;  2677          4208  1 %SBTTL 'INIT_ATTR - initialize attributes area'
;  2678          4209  1 GLOBAL ROUTINE INIT_ATTR(INPUT_HDR): NOVALUE=
;  2679          4210  1
;  2680          4211  1 !++
;  2681          4212  1 !
;  2682          4213  1 ! FUNCTIONAL DESCRIPTION:
;  2683          4214  1 !       This routine initializes the structure-independent attributes area.
;  2684          4215  1 !
;  2685          4216  1 ! INPUT PARAMETERS:
;  2686          4217  1 !       INPUT_HDR       - Input file header.
;  2687          4218  1 !
;  2688          4219  1 ! IMPLICIT INPUTS:
;  2689          4220  1 !       NONE
;  2690          4221  1 !
;  2691          4222  1 ! OUTPUT PARAMETERS:
;  2692          4223  1 !       NONE
;  2693          4224  1 !
;  2694          4225  1 ! IMPLICIT OUTPUTS:
;  2695          4226  1 !       INPUT_CREDATE, INPUT_REVDATE, INPUT_EXPDATE, INPUT_BAKDATE,
;  2696          4227  1 !       INPUT_FILEOWNER, INPUT_FILECHAR, INPUT_RECATTR
;  2697          4228  1 !                       - The input attributes.
;  2698          4229  1 !
;  2699          4230  1 ! ROUTINE VALUE:
;  2700          4231  1 !       NONE
;  2701          4232  1 !
;  2702          4233  1 ! SIDE EFFECTS:
;  2703          4234  1 !       NONE
;  2704          4235  1 !
;  2705          4236  1 !--
;  2706          4237  1
;  2707          4238  2 BEGIN
;  2708          4239  2 MAP
;  2709          4240  2         INPUT_HDR:      REF BBLOCK;     ! Pointer to file header
;  2710          4241  2 LOCAL
;  2711          4242  2         INPUT_IDENT:    REF BBLOCK,     ! Pointer to ident area
;  2712          4243  2         IDLEN;                          ! Length of ODS-2 ident area
;  2713          4244  2 MACRO
;  2714        M 4245  2         MOVQ_(A,B)=
;  2715        M 4246  2             (B) = .(A);
;  2716          4247  2             ((B)+4) = .((A)+4) %;
;  2717          4248  2
;  2718          4249  2
;  2719          4250  2 CH$FILL(0, INPUT_HDR_END-INPUT_HDR_BEG, INPUT_HDR_BEG);
;  2720          4251  2
;  2721          4252  2
;  2722          4253  2 IF .INPUT_HDR[FH2$B_STRUCLEV] EQL 1
;  2723          4254  2 THEN
;  2724          4255  3     BEGIN
;  2725          4256  3
;  2726          4257  3     ! Determine the address of the ident area.
;  2727          4258  3     !
;  2728          4259  3     INPUT_IDENT = .INPUT_HDR + .INPUT_HDR[FH1$B_IDOFFSET] * 2;
;  2729          4260  3
;  2730          4261  3
;  2731          4262  3     ! Initialize the structure-independent attributes area.
;  2732          4263  3     !
;  2733          4264  3     FROM_ODS1_DATE(INPUT_IDENT[FI1$T_CREDATE], INPUT_CREDATE);
```

```
: 2734          4265  3     FROM_ODS1_DATE(INPUT_IDENT[FI1$T_REVDATE], INPUT_REVDATE);
: 2735          4266  3     FROM_ODS1_DATE(INPUT_IDENT[FI1$T_EXPDATE], INPUT_EXPDATE);
: 2736          4267  3     INPUT_FILEOWNER< 0,16> = .INPUT_HDR[FH1$B_UICMEMBER];
: 2737          4268  3     INPUT_FILEOWNER<16,16> = .INPUT_HDR[FH1$B_UICGROUP];
: 2738          4269  3     INPUT_FILECHAR = .INPUT_HDR[FH1$W_FILECHAR];
: 2739          4270  3     CH$MOVE(FAT$C_LENGTH, INPUT_HDR[FA1$W_RECATTR], INPUT_RECATTR);
: 2740          4271  3     END
: 2741          4272  2 ELSE
: 2742          4273  3     BEGIN
: 2743          4274  3
: 2744          4275  3     ! Determine the address and length of the ident area.
: 2745          4276  3     !
: 2746          4277  3     INPUT_IDENT = .INPUT_HDR + .INPUT_HDR[FH2$B_IDOFFSET] * 2;
: 2747          4278  3     IDLEN = (.INPUT_HDR[FH2$B_MPOFFSET] - .INPUT_HDR[FH2$B_IDOFFSET]) * 2;
: 2748          4279  3
: 2749          4280  3
: 2750          4281  3     ! Initialize the structure-independent attributes area.
: 2751          4282  3     !
: 2752          4283  3     IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_CREDATE)+8
: 2753          4284  3     THEN
: 2754          4285  4         BEGIN
: 2755          4286  4         MOVQ_(INPUT_IDENT[FI2$Q_CREDATE], INPUT_CREDATE);
: 2756          4287  3         END;
: 2757          4288  3     IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_REVDATE)+8
: 2758          4289  3     THEN
: 2759          4290  4         BEGIN
: 2760          4291  4         MOVQ_(INPUT_IDENT[FI2$Q_REVDATE], INPUT_REVDATE);
: 2761          4292  3         END;
: 2762          4293  3     IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_EXPDATE)+8
: 2763          4294  3     THEN
: 2764          4295  4         BEGIN
: 2765          4296  4         MOVQ_(INPUT_IDENT[FI2$Q_EXPDATE], INPUT_EXPDATE);
: 2766          4297  3         END;
: 2767          4298  3     IF .IDLEN GEQU $BYTEOFFSET(FI2$Q_BAKDATE)+8
: 2768          4299  3     THEN
: 2769          4300  4         BEGIN
: 2770          4301  4         MOVQ_(INPUT_IDENT[FI2$Q_BAKDATE], INPUT_BAKDATE);
: 2771          4302  3         END;
: 2772          4303  3     INPUT_FILEOWNER = .INPUT_HDR[FH2$L_FILEOWNER];
: 2773          4304  3     INPUT_FILECHAR = .INPUT_HDR[FH2$L_FILECHAR];
: 2774          4305  3     CH$MOVE(FAT$C_LENGTH, INPUT_HDR[FA2$W_RECATTR], INPUT_RECATTR);
: 2775          4306  2     END;
: 2776          4307  1 END;


                                                03FC 00000          .ENTRY  INIT_ATTR, Save R2,R3,R4,R5,R6,R7,R8,R9     : 4209
                                    59     F1C0 CF  9E 00002        MOVAB   FROM_ODS1_DATE, R9                          :
                                    58 00000000' EF  9E 00007        MOVAB   INPUT_HDR_BEG, R8                           :
0048    8F              00          6E          00  2C 0000E        MOVC5   #0, (SP), #0, #72, INPUT_HDR_BEG            : 4250
                                                68      00015                                                           :
                                    57       04 AC  D0 00016        MOVL    INPUT_HDR, R7                               : 4253
                                    01       07 A7  91 0001A        CMPB    7(R7), #1                                   :
                                                37  12 0001E        BNEQ    1$                                          :
                                    50             67  9A 00020        MOVZBL  (R7), R0                                : 4259
```

```
                                56          6740  3E 00023          MOVAW   (R7)[R0], INPUT_IDENT                   ;
                                   58          DD 00027          PUSHL   R8                                      ; 4264
                                19 A6          9F 00029          PUSHAB  25(INPUT_IDENT)                         ;
                                69             02 FB 0002C       CALLS   #2, FROM_ODS1_DATE                      ;
                                08 A8          9F 0002F          PUSHAB  INPUT_REVDATE                           ; 4265
                                0C A6          9F 00032          PUSHAB  12(INPUT_IDENT)                         ;
                                69             02 FB 00035       CALLS   #2, FROM_ODS1_DATE                      ;
                                10 A8          9F 00038          PUSHAB  INPUT_EXPDATE                           ; 4266
                                26 A6          9F 0003B          PUSHAB  38(INPUT_IDENT)                         ;
                                69             02 FB 0003E       CALLS   #2, FROM_ODS1_DATE                      ;
                          20 A8 08 A7          9B 00041          MOVZBW  8(R7), INPUT_FILEOWNER                  ; 4267
                          22 A8 09 A7          9B 00046          MOVZBW  9(R7), INPUT_FILEOWNER+2                ; 4268
                          24 A8 0C A7          3C 0004B          MOVZWL  12(R7), INPUT_FILECHAR                  ; 4269
                    28 A8 0E A7                20 28 00050       MOVC3   #32, 14(R7), INPUT_RECATTR              ; 4270
                                                04 00056          RET                                             ; 4253
                                50             67 9A 00057 1$:   MOVZBL  (R7), R0                                ; 4277
                                56          6740  3E 0005A          MOVAW   (R7)[R0], INPUT_IDENT                   ;
                                51 01 A7       9A 0005E          MOVZBL  1(R7), R1                               ; 4278
                             50 51             50 C3 00062       SUBL3   R0, R1, R0                              ;
                                50             02 C4 00066       MULL2   #2, IDLEN                               ;
                                1E             50 D1 00069       CMPL    IDLEN, #30                              ; 4283
                                               04 1F 0006C       BLSSU   2$                                      ;
                                68 16 A6       7D 0006E          MOVQ    22(INPUT_IDENT), INPUT_CREDATE          ; 4286
                                26             50 D1 00072 2$:   CMPL    IDLEN, #38                              ; 4288
                                               05 1F 00075       BLSSU   3$                                      ;
                          08 A8 1E A6          7D 00077          MOVQ    30(INPUT_IDENT), INPUT_REVDATE          ; 4291
                                2E             50 D1 0007C 3$:   CMPL    IDLEN, #46                              ; 4293
                                               05 1F 0007F       BLSSU   4$                                      ;
                          10 A8 26 A6          7D 00081          MOVQ    38(INPUT_IDENT), INPUT_EXPDATE          ; 4296
                                36             50 D1 00086 4$:   CMPL    IDLEN, #54                              ; 4298
                                               05 1F 00089       BLSSU   5$                                      ;
                          18 A8 2E A6          7D 0008B          MOVQ    46(INPUT_IDENT), INPUT_BAKDATE          ; 4301
                          20 A8 3C A7          D0 00090 5$:      MOVL    60(R7), INPUT_FILEOWNER                 ; 4303
                          24 A8 34 A7          D0 00095          MOVL    52(R7), INPUT_FILECHAR                  ; 4304
                    28 A8 14 A7                20 28 0009A       MOVC3   #32, 20(R7), INPUT_RECATTR              ; 4305
                                                04 000A0          RET                                             ; 4307

; Routine Size:  161 bytes,    Routine Base:  CODE + 14B0
```

```
;  2778     4308  1 %SBTTL 'SELECT_INPUT_FILE - apply selection to input file'
;  2779     4309  1 GLOBAL ROUTINE SELECT_INPUT_FILE(TYPE)=
;  2780     4310  1
;  2781     4311  1 !++
;  2782     4312  1 !
;  2783     4313  1 ! FUNCTIONAL DESCRIPTION:
;  2784     4314  1 !       This routine evaluates selection criteria for an input file.
;  2785     4315  1 !
;  2786     4316  1 ! INPUT PARAMETERS:
;  2787     4317  1 !       TYPE            - <0,1> to evaluate file header criteria.
;  2788     4318  1 !                         <1,1> to evaluate file name criteria.
;  2789     4319  1 !                         <2,1> to evaluate /CONFIRM qualifier.
;  2790     4320  1 !
;  2791     4321  1 ! IMPLICIT INPUTS:
;  2792     4322  1 !       INPUT_NAM       - pointer to the NAM block.
;  2793     4323  1 !       INPUT_CREDATE, INPUT_REVDATE, INPUT_EXPDATE, INPUT_BAKDATE,
;  2794     4324  1 !       INPUT_FILEOWNER, INPUT_FILECHAR, INPUT_RECATTR
;  2795     4325  1 !                       - The input attributes.
;  2796     4326  1 !
;  2797     4327  1 ! OUTPUT PARAMETERS:
;  2798     4328  1 !       NONE
;  2799     4329  1 !
;  2800     4330  1 ! IMPLICIT OUTPUTS:
;  2801     4331  1 !       NONE
;  2802     4332  1 !
;  2803     4333  1 ! ROUTINE VALUE:
;  2804     4334  1 !       True if file is selected by the selection qualifiers, false otherwise.
;  2805     4335  1 !
;  2806     4336  1 ! SIDE EFFECTS:
;  2807     4337  1 !       NONE
;  2808     4338  1 !
;  2809     4339  1 !--
;  2810     4340  1
;  2811     4341  2 BEGIN
;  2812     4342  2 LOCAL
;  2813     4343  2         CMDDATE:        REF VECTOR,     ! Pointer to command selection date
;  2814     4344  2         FILDATE:        REF VECTOR;     ! Pointer to file selection date
;  2815     4345  2
;  2816     4346  2
;  2817     4347  2 ! For a save operation, avoid applying file selection qualifiers to ensure that
;  2818     4348  2 ! all needed directories are saved.  This test is disabled under /INTERCHANGE
;  2819     4349  2 ! to save space on distribution media.
;  2820     4350  2 !
;  2821     4351  2 IF .QUAL[QUAL_OSAV] AND .DIR_STATUS[D_STAT_VALID] AND NOT .QUAL[QUAL_INTE]
;  2822     4352  2 THEN
;  2823     4353  2     RETURN TRUE;
;  2824     4354  2
;  2825     4355  2
;  2826     4356  2 IF .TYPE<0,1>
;  2827     4357  2 THEN
;  2828     4358  3     BEGIN
;  2829     4359  3     IF .QUAL[QUAL_BEFO] OR .QUAL[QUAL_SINC]
;  2830     4360  3     THEN
;  2831     4361  4         BEGIN
;  2832     4362  4         IF .QUAL[QUAL_BACK] THEN FILDATE = INPUT_BAKDATE ELSE
;  2833     4363  4         IF .QUAL[QUAL_CREA] THEN FILDATE = INPUT_CREDATE ELSE
;  2834     4364  4         IF .QUAL[QUAL_EXPI] THEN FILDATE = INPUT_EXPDATE ELSE
```

```
; 2835          4365  4                                   FILDATE = INPUT_REVDATE;
; 2836          4366  3          END;
; 2837          4367  3
; 2838          4368  3
; 2839          4369  3      ! BEFORE
; 2840          4370  3      !
; 2841          4371  3      IF .QUAL[QUAL_BEFO]
; 2842          4372  3      THEN
; 2843          4373  4          BEGIN
; 2844          4374  4          IF .QUAL[QUAL_BEFO_BACK]
; 2845          4375  4              THEN CMDDATE = INPUT_BAKDATE
; 2846          4376  4              ELSE CMDDATE = QUAL[QUAL_BEFO_VALUE];
; 2847          4377  4          IF .FILDATE[1] GTRU .CMDDATE[1] THEN RETURN FALSE;
; 2848          4378  4          IF .FILDATE[1] EQLU .CMDDATE[1] AND .FILDATE[0] GEQU .CMDDATE[0] THEN RETURN FALSE;
; 2849          4379  3          END;
; 2850          4380  3
; 2851          4381  3
; 2852          4382  3      ! OWNER
; 2853          4383  3      !
; 2854          4384  3      IF .QUAL[QUAL_I_OWNE]
; 2855          4385  3      THEN
; 2856          4386  4          IF NOT (
; 2857          4387  4              (.QUAL[QUAL_I_OWN_WGRP] OR .INPUT_FILEOWNER<16,16> EQL .QUAL[QUAL_I_OWN_GRP]) AND
; 2858          4388  4              (.QUAL[QUAL_I_OWN_WMEM] OR .INPUT_FILEOWNER< 0,16> EQL .QUAL[QUAL_I_OWN_MEM]))
; 2859          4389  3          THEN
; 2860          4390  3              RETURN FALSE;
; 2861          4391  3
; 2862          4392  3
; 2863          4393  3      ! SINCE
; 2864          4394  3      !
; 2865          4395  3      IF .QUAL[QUAL_SINC]
; 2866          4396  3      THEN
; 2867          4397  4          BEGIN
; 2868          4398  4          IF .QUAL[QUAL_SINC_BACK]
; 2869          4399  4              THEN CMDDATE = INPUT_BAKDATE
; 2870          4400  4              ELSE CMDDATE = QUAL[QUAL_SINC_VALUE];
; 2871          4401  4          IF .FILDATE[1] LSSU .CMDDATE[1] THEN RETURN FALSE;
; 2872          4402  4          IF .FILDATE[1] EQLU .CMDDATE[1] AND .FILDATE[0] LSSU .CMDDATE[0] THEN RETURN FALSE;
; 2873          4403  3          END;
; 2874          4404  2      END;
; 2875          4405  2
; 2876          4406  2
; 2877          4407  2  IF .TYPE<1,1>
; 2878          4408  2  THEN
; 2879          4409  3      BEGIN
; 2880          4410  3
; 2881          4411  3      ! EXCLUDE
; 2882          4412  3      !
; 2883          4413  3      IF .QUAL[QUAL_EXCL]
; 2884          4414  3      THEN
; 2885          4415  4          BEGIN
; 2886          4416  4          LOCAL
; 2887          4417  4              RSA_DESC:   VECTOR[2],
; 2888          4418  4              EXC[:       REF BBLOCK;
; 2889          4419  4
; 2890          4420  4          RSA_DESC[0] = .INPUT_NAM[NAM$B_RSL];
; 2891          4421  4          RSA_DESC[1] = .INPUT_NAM[NAM$L_RSA];
```

```
;2892            4422  4          EXCL = .QUAL[QUAL_EXCL_LIST];
;2893            4423  4          WHILE .EXCL NEQ 0 DO
;2894            4424  5              BEGIN
;2895            4425  5              IF MATCH(RSA_DESC, EXCL[QUAL_EXCL_DESC]) THEN RETURN FALSE;
;2896            4426  5              EXCL = .EXCL[QUAL_NEXT];
;2897            4427  4              END;
;2898            4428  3          END;
;2899            4429  2      END;
;2900            4430  2
;2901            4431  2
;2902            4432  2 IF .TYPE<2,1>
;2903            4433  2 THEN
;2904            4434  3      BEGIN
;2905            4435  3
;2906            4436  3      ! CONFIRM
;2907            4437  3      !
;2908            4438  3      IF .QUAL[QUAL_CONF]
;2909            4439  3      THEN
;2910            4440  4          BEGIN
;2911            4441  4          LOCAL
;2912            4442  4              FAO_BUFFER: VECTOR[256,BYTE],
;2913            4443  4              FAO_DESC:   BBLOCK[8],
;2914            4444  4              ANS_BUFFER: VECTOR[8,BYTE],
;2915            4445  4              ANS_DESC:   BBLOCK[8];
;2916            4446  4
;2917            4447  4          FAO_DESC[DSC$W_LENGTH] = 256;
;2918            4448  4          FAO_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;2919            4449  4          FAO_DESC[DSC$B_CLASS] = DSC$K_CLASS_S;
;2920            4450  4          FAO_DESC[DSC$A_POINTER] = FAO_BUFFER;
;2921            4451  4          ANS_DESC[DSC$W_LENGTH] = 8;
;2922            4452  4          ANS_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;2923            4453  4          ANS_DESC[DSC$B_CLASS] = DSC$K_CLASS_S;
;2924            4454  4          ANS_DESC[DSC$A_POINTER] = ANS_BUFFER;
;2925          P 4455  4          $FAO(
;2926          P 4456  4              (IF .QUAL[QUAL_COMP]
;2927          P 4457  4                  THEN $DESCRIPTOR('!AD, compare? (Y or N): ')
;2928          P 4458  4                  ELSE $DESCRIPTOR('!AD, copy? (Y or N): ')),
;2929          P 4459  4              FAO_DESC,
;2930          P 4460  4              FAO_DESC,
;2931          P 4461  4              .INPUT_NAM[NAM$B_RSL],
;2932            4462  4              .INPUT_NAM[NAM$L_RSA]);
;2933            4463  4          LIB$GET_COMMAND(ANS_DESC, FAO_DESC);
;2934            4464  4          IF .ANS_BUFFER[0] NEQ %C'Y' AND .ANS_BUFFER[0] NEQ %C'y' THEN RETURN FALSE;
;2935            4465  3          END;
;2936            4466  2      END;
;2937            4467  2
;2938            4468  2
;2939            4469  2 ! Passes all tests, return true.
;2940            4470  2 !
;2941            4471  2 TRUE
;2942            4472  1 END;


28  20  3F  65  72  61  70  6D  6F  63  20  2C  44  41  21  01551 P.AAE:  .ASCII  \!AD, compare? (Y or N): \                    ;
                        20  3A  29  4E  20  72  6F  20  59  01560                                                            ;
                                                            01569        .BLKB   3
```

```
                                         00000018  0156C P.AAD:   .LONG    24                                  ;
                                         00000000' 01570          .ADDRESS P.AAE                               ;
6F 20 59 28 20 3F 79 70 6F 63 20 2C 44 41 21      01574 P.AAG:   .ASCII   \!AD, copy? (Y or N): \             ;
                              20 3A 29 4E 20 72   01583                                                        ;
                                                  01589          .BLKB    3
                                         00000015  0158C P.AAF:   .LONG    21                                  ;
                                         00000000' 01590          .ADDRESS P.AAG                               ;

                                                                  .EXTRN   SYS$FAO

                                             000C 00000          .ENTRY   SELECT_INPUT_FILE, Save R2,R3       ; 4309
                         53 00000000'   EF  9E 00002          MOVAB    QUAL+8, R3                          ;
                         5E       FEE8  CE  9E 00009          MOVAB    -280(SP), SP                        ;
                                    07  A3  95 0000E          TSTB     QUAL+15                             ; 4351
                                        0D  18 00011          BGEQ     1$                                  ;
                         08       0297  C3  E9 00013          BLBC     DIR_STATUS, 1$                      ;
          03       06    A3             01  E0 00018          BBS      #1, QUAL+14, 1$                     ;
                                      013D  31 0001D          BRW      22$                                 ;
                         03         04  AC  E8 00020 1$:      BLBS     TYPE, 2$                            ; 4356
                                      0097  31 00024          BRW      16$                                 ;
          05             63             02  E0 00027 2$:      BBS      #2, QUAL+8, 3$                      ; 4359
          28       05    A3             04  E1 0002B          BBC      #4, QUAL+13, 7$                     ;
          07             63             01  E1 00030 3$:      BBC      #1, QUAL+8, 4$                      ; 4362
                         51       010C  C3  9E 00034          MOVAB    INPUT_BAKDATE, FILDATE              ;
                                        1D  11 00039          BRB      7$                                  ;
          07       01    A3             02  E1 0003B 4$:      BBC      #2, QUAL+9, 5$                      ; 4363
                         51       00F4  C3  9E 00040          MOVAB    INPUT_CREDATE, FILDATE              ;
                                        11  11 00045          BRB      7$                                  ;
          07       01    A3             05  E1 00047 5$:      BBC      #5, QUAL+9, 6$                      ; 4364
                         51       0104  C3  9E 0004C          MOVAB    INPUT_EXPDATE, FILDATE              ;
                                        05  11 00051          BRB      7$                                  ;
                         51       00FC  C3  9E 00053 6$:      MOVAB    INPUT_REVDATE, FILDATE              ; 4365
          20             63             02  E1 00058 7$:      BBC      #2, QUAL+8, 11$                     ; 4371
          07             63             03  E1 0005C          BBC      #3, QUAL+8, 8$                      ; 4374
                         50       010C  C3  9E 00060          MOVAB    INPUT_BAKDATE, CMDDATE              ; 4375
                                        04  11 00065          BRB      9$                                  ;
                         50         08  A3  9E 00067 8$:      MOVAB    QUAL+16, CMDDATE                    ; 4376
                   04    A0         04  A1  D1 0006B 9$:      CMPL     4(FILDATE), 4(CMDDATE)              ; 4377
                                        07  1A 00070          BGTRU    10$                                 ;
                                        08  12 00072          BNEQ     11$                                 ; 4378
                         60             61  D1 00074          CMPL     (FILDATE), (CMDDATE)                ;
                                        03  1F 00077          BLSSU    11$                                 ;
                                      00DE  31 00079 10$:     BRW      21$                                 ;
          1A       03    A3             05  E1 0007C 11$:     BBC      #5, QUAL+11, 13$                    ; 4384
          08       03    A3             06  E0 00081          BBS      #6, QUAL+11, 12$                    ; 4387
                   36    A3       0116  C3  B1 00086          CMPW     INPUT_FILEOWNER+2, QUAL+62          ;
                                        EB  12 0008C          BNEQ     10$                                 ;
                                    03  A3  95 0008E 12$:     TSTB     QUAL+11                             ; 4388
                                        08  19 00091          BLSS     13$                                 ;
                   34    A3       0114  C3  B1 00093          CMPW     INPUT_FILEOWNER, QUAL+60            ;
                                        DE  12 00099          BNEQ     10$                                 ;
          1E       05    A3             04  E1 0009B 13$:     BBC      #4, QUAL+13, 16$                    ; 4395
          07       05    A3             05  E1 000A0          BBC      #5, QUAL+13, 14$                    ; 4398
                         50       010C  C3  9E 000A5          MOVAB    INPUT_BAKDATE, CMDDATE              ; 4399
                                        04  11 000AA          BRB      15$                                 ;
                         50         18  A3  9E 000AC 14$:     MOVAB    QUAL+32, CMDDATE                    ; 4400
                   04    A0         04  A1  D1 000B0 15$:     CMPL     4(FILDATE), 4(CMDDATE)              ; 4401
```

```
                                        C2  1F 000B5          BLSSU   10$                                                    ;
                                        05  12 000B7          BNEQ    16$                                                    ; 4402
                                60      61  D1 000B9          CMPL    (FILDATE), (CMDDATE)                                   ;
                                        BB  1F 000BC          BLSSU   10$                                                    ;
                    2F      04  AC      01  E1 000BE 16$:      BBC     #1, TYPE, 18$                                          ; 4407
                    2A      01  A3      04  E1 000C3          BBC     #4, QUAL+9, 18$                                        ; 4413
                                50  00C8  C3  D0 000C8          MOVL    INPUT_NAM, R0                                          ; 4420
                            F8  AD  03  A0  9A 000CD          MOVZBL  3(R0), RSA_DESC                                        ;
                            FC  AD  04  A0  D0 000D2          MOVL    4(R0), RSA_DESC+4                                      ; 4421
                                52  28  A3  D0 000D7          MOVL    QUAL+48, EXCL                                          ; 4422
                                        15  13 000DB 17$:      BEQL    18$                                                    ; 4423
                                    04  A2  9F 000DD          PUSHAB  4(EXCL)                                                ; 4425
                                    F8  AD  9F 000E0          PUSHAB  RSA_DESC                                               ;
                00000000G       00      02  FB 000E3          CALLS   #2, MATCH                                              ;
                                6D      50  E8 000EA          BLBS    R0, 21$                                                ;
                                52      62  D0 000ED          MOVL    (EXCL), EXCL                                           ; 4426
                                        E9  11 000F0          BRB     17$                                                    ; 4423
                    66      04  AC      02  E1 000F2 18$:      BBC     #2, TYPE, 22$                                          ; 4432
                                62  01  A3  E9 000F7          BLBC    QUAL+9, 22$                                            ; 4438
                            10  AE  010E0100  8F  D0 000FB      MOVL    #17694976, FAO_DESC                                    ; 4447
                            14  AE  18  AE  9E 00103          MOVAB   FAO_BUFFER, FAO_DESC+4                                 ; 4450
                                6E  010E0008  8F  D0 00108      MOVL    #17694728, ANS_DESC                                    ; 4451
                            04  AE  08  AE  9E 0010F          MOVAB   ANS_BUFFER, ANS_DESC+4                                 ; 4454
                                50  00C8  C3  D0 00114          MOVL    INPUT_NAM, R0                                          ; 4462
                                    04  A0  DD 00119          PUSHL   4(R0)                                                  ;
                                7E      03  A0  9A 0011C      MOVZBL  3(R0), -(SP)                                           ;
                                    18  AE  9F 00120          PUSHAB  FAO_DESC                                               ;
                                    1C  AE  9F 00123          PUSHAB  FAO_DESC                                               ;
                                        63  95 00126          TSTB    QUAL+8                                                 ;
                                        07  18 00128          BGEQ    19$                                                    ;
                                50  FEAA  CF  9E 0012A          MOVAB   P.AAD, R0                                              ;
                                        05  11 0012F          BRB     20$                                                    ;
                                50  FEC3  CF  9E 00131 19$:      MOVAB   P.AAF, R0                                              ;
                                        50  DD 00136 20$:      PUSHL   R0                                                     ;
                00000000G       00      05  FB 00138          CALLS   #5, SYS$FAO                                            ;
                                    10  AE  9F 0013F          PUSHAB  FAO_DESC                                               ; 4463
                                    04  AE  9F 00142          PUSHAB  ANS_DESC                                               ;
                00000000G       00      02  FB 00145          CALLS   #2, LIB$GET_COMMAND                                    ;
                            59  8F  08  AE  91 0014C          CMPB    ANS_BUFFER, #89                                        ; 4464
                                        0A  13 00151          BEQL    22$                                                    ;
                            79  8F  08  AE  91 00153          CMPB    ANS_BUFFER, #121                                       ;
                                        03  13 00158          BEQL    22$                                                    ;
                                        50  D4 0015A 21$:      CLRL    R0                                                     ;
                                            04 0015C          RET                                                            ;
                                50      01  D0 0015D 22$:      MOVL    #1, R0                                                 ; 4472
                                            04 00160          RET                                                            ;

; Routine Size:  353 bytes,    Routine Base:  CODE + 1594
```

```
;2944           4473  1 %SBTTL 'POSTPROCESS_FILES - perform file postprocessing'
;2945           4474  1 ROUTINE POSTPROCESS_FILES: NOVALUE=
;2946           4475  1
;2947           4476  1 !++
;2948           4477  1 !
;2949           4478  1 ! FUNCTIONAL DESCRIPTION:
;2950           4479  1 !       This routine records the backup date (/RECORD) or deletes (/DELETE)
;2951           4480  1 !       each file successfully processed.
;2952           4481  1 !
;2953           4482  1 ! INPUT PARAMETERS:
;2954           4483  1 !       NONE
;2955           4484  1 !
;2956           4485  1 ! IMPLICIT INPUTS:
;2957           4486  1 !       INPUT_PROC_LIST - Pointer to head of list.
;2958           4487  1 !
;2959           4488  1 ! OUTPUT PARAMETERS:
;2960           4489  1 !       NONE
;2961           4490  1 !
;2962           4491  1 ! IMPLICIT OUTPUTS:
;2963           4492  1 !       NONE
;2964           4493  1 !
;2965           4494  1 ! ROUTINE VALUE:
;2966           4495  1 !       NONE
;2967           4496  1 !
;2968           4497  1 ! SIDE EFFECTS:
;2969           4498  1 !       Backup date recorded or file deleted.
;2970           4499  1 !
;2971           4500  1 !--
;2972           4501  1
;2973           4502  2 BEGIN
;2974           4503  2 LOCAL
;2975           4504  2         IN_QUAL:        REF BBLOCK,     ! Pointer to current input qualifiers
;2976           4505  2         P:              REF BBLOCK,     ! Pointer to file ID block
;2977           4506  2         Q:              REF BBLOCK,     ! Pointer to file ID within block
;2978           4507  2         FIB:            BBLOCK[FIB$C_LENGTH],   ! FIB
;2979           4508  2         FIB_DESC:       VECTOR[2],      ! Descriptor for FIB
;2980           4509  2         STATUS,                         ! Status variable
;2981           4510  2         IOSB:           VECTOR[4,WORD], ! I/O status block
;2982           4511  2         WRITELOCKED;                    ! True if input volume writelocked
;2983           4512  2
;2984           4513  2
;2985           4514  2 ! Initialize.
;2986           4515  2 !
;2987           4516  2 IN_QUAL = 0;
;2988           4517  2 WRITELOCKED = FALSE;
;2989           4518  2 FIB_DESC[0] = FIB$C_LENGTH;
;2990           4519  2 FIB_DESC[1] = FIB;
;2991           4520  2 IF .QUAL[QUAL_DELE]
;2992           4521  2     THEN SIGNAL(BACKUP$_STARTDELETE)
;2993           4522  2     ELSE SIGNAL(BACKUP$_STARTRECORD);
;2994           4523  2
;2995           4524  2
;2996           4525  2 ! Loop until file ID block list is empty.
;2997           4526  2 !
;2998           4527  2 WHILE .INPUT_PROC_LIST NEQ 0 DO
;2999           4528  3     BEGIN
;3000           4529  3
```

```
;  3001          4530  3      ! Remove the first file ID block from the list.
;  3002          4531  3      !
;  3003          4532  3      P = .INPUT_PROC_LIST;
;  3004          4533  3      INPUT_PROC_LIST = .P[REC_NEXT];
;  3005          4534  3
;  3006          4535  3
;  3007          4536  3      ! Reassign the channel if necessary.
;  3008          4537  3      !
;  3009          4538  3      IF .P[REC_QUAL] NEQ .IN_QUAL
;  3010          4539  3      THEN
;  3011          4540  4          BEGIN
;  3012          4541  4
;  3013          4542  4          ! Point to the current qualifiers block.
;  3014          4543  4          !
;  3015          4544  4          IN_QUAL = .P[REC_QUAL];
;  3016          4545  4
;  3017          4546  4
;  3018          4547  4          ! Assign the channel.
;  3019          4548  4          !
;  3020          4549  4          IF .INPUT_CHAN NEQ 0 THEN $DASSGN(CHAN=.INPUT_CHAN);
;  3021          4550  4          STATUS = ASSIGN_INPUT_CHANNEL(IN_QUAL[QUAL_DEV_DESC], INPUT_CHAN, 0, 0);
;  3022          4551  4          IF NOT .STATUS
;  3023          4552  4          THEN
;  3024          4553  4              SIGNAL(
;  3025          4554  4                  BACKUP$_OPENOUT + STS$K_SEVERE, 1, IN_QUAL[QUAL_DEV_DESC],
;  3026          4555  4                  .STATUS);
;  3027          4556  4          WRITELOCKED = FALSE;
;  3028          4557  3          END;
;  3029          4558  3
;  3030          4559  3
;  3031          4560  3      ! Loop over all entries within the block.
;  3032          4561  3      !
;  3033          4562  3      IF NOT .WRITELOCKED
;  3034          4563  3      THEN
;  3035          4564  4          BEGIN
;  3036          4565  4          Q = P[REC_FID_BASE];
;  3037          4566  4          DECR I FROM .P[REC_USED] TO 1 DO
;  3038          4567  5              BEGIN
;  3039          4568  5              CH$FILL(0, FIB$C_LENGTH, FIB);
;  3040          4569  5              FIB[FIB$W_FID_NUM] = .BBLOCK[Q[REC_FID], FID$W_NUM];
;  3041          4570  5              FIB[FIB$W_FID_SEQ] = .BBLOCK[Q[REC_FID], FID$W_SEQ];
;  3042          4571  5              FIB[FIB$W_FID_RVN] = .BBLOCK[Q[REC_FID], FID$W_RVN];
;  3043          4572  5
;  3044          4573  5
;  3045          4574  5              IF .QUAL[QUAL_DELE]
;  3046          4575  5              THEN
;  3047          4576  6                  BEGIN
;  3048          4577  6                  FIB[FIB$W_DID_NUM] = .BBLOCK[Q[REC_DID], FID$W_NUM];
;  3049          4578  6                  FIB[FIB$W_DID_SEQ] = .BBLOCK[Q[REC_DID], FID$W_SEQ];
;  3050          4579  6                  FIB[FIB$W_DID_RVN] = .BBLOCK[Q[REC_DID], FID$W_RVN];
;  3051          4580  6                  FIB[FIB$V_FINDFID] = TRUE;
;  3052          4581  6
;  3053          4582  6
;  3054          4583  6                  ! Issue the delete QIO to delete the specified file.
;  3055          4584  6                  !
;  3056        P 4585  6                  STATUS = $QIOW(
;  3057        P 4586  6                      FUNC=IO$_DELETE OR IO$M_DELETE,
```

```
;3058      P 4587  6                    CHAN=.INPUT_CHAN,
;3059      P 4588  6                    IOSB=IOSB,
;3060        4589  6                    P1=FIB_DESC);
;3061        4590  6                IF .STATUS THEN STATUS = .IOSB[0];
;3062        4591  6                IF NOT .STATUS
;3063        4592  6                THEN
;3064        4593  6                    IF .STATUS<0,16> EQL SS$_WRITLCK
;3065        4594  6                    THEN
;3066        4595  7                        BEGIN
;3067        4596  7                        SIGNAL(BACKUP$_DELETESWL, 1, IN_QUAL[QUAL_DEV_DESC]);
;3068        4597  7                        WRITELOCKED = TRUE;
;3069        4598  7                        EXITLOOP;
;3070        4599  7                        END
;3071        4600  6                    ELSE
;3072        4601  6                        SIGNAL(
;3073        4602  6                            BACKUP$_DELETE,
;3074        4603  6                            4,
;3075        4604  6                            IN_QUAL[QUAL_DEV_DESC],
;3076        4605  6                            .FIB[FIB$W_FID_NUM] + .FIB[FIB$B_FID_NMX] ^ 16,
;3077        4606  6                            .FIB[FIB$W_FID_SEQ],
;3078        4607  6                            .FIB[FIB$B_FID_RVN],
;3079        4608  6                            .STATUS);
;3080        4609  6                END
;3081        4610  5            ELSE
;3082        4611  6                BEGIN
;3083        4612  6
;3084        4613  6                ! Issue the modify QIO to write the backup date of the specified
;3085        4614  6                ! file.
;3086        4615  6                !
;3087        4616  6                FIB[FIB$V_NORECORD] = TRUE;
;3088      P 4617  6                STATUS = $QIOW(
;3089      P 4618  6                    FUNC=IO$_MODIFY,
;3090      P 4619  6                    CHAN=.INPUT_CHAN,
;3091      P 4620  6                    IOSB=IOSB,
;3092      P 4621  6                    P1=FIB_DESC,
;3093      P 4622  6                    P5=UPLIT(
;3094      P 4623  6                        WORD(ATR$S_BAKDATE, ATR$C_BAKDATE),
;3095      P 4624  6                        LONG(JPI_DATE),
;3096        4625  6                        LONG(0))];
;3097        4626  6                IF .STATUS THEN STATUS = .IOSB[0];
;3098        4627  6                IF NOT .STATUS
;3099        4628  6                THEN
;3100        4629  6                    IF .STATUS<0,16> EQL SS$_WRITLCK
;3101        4630  6                    THEN
;3102        4631  7                        BEGIN
;3103        4632  7                        SIGNAL(BACKUP$_RECORDSWL, 1, IN_QUAL[QUAL_DEV_DESC]);
;3104        4633  7                        WRITELOCKED = TRUE;
;3105        4634  7                        EXITLOOP;
;3106        4635  7                        END
;3107        4636  6                    ELSE
;3108        4637  6                        SIGNAL(
;3109        4638  6                            BACKUP$_WRITEBACK,
;3110        4639  6                            4,
;3111        4640  6                            IN_QUAL[QUAL_DEV_DESC],
;3112        4641  6                            .FIB[FIB$W_FID_NUM] + .FIB[FIB$B_FID_NMX] ^ 16,
;3113        4642  6                            .FIB[FIB$W_FID_SEQ],
;3114        4643  6                            .FIB[FIB$B_FID_RVN],
```

```
;3115          4644  6                                .STATUS);
;3116          4645  5                    END;
;3117          4646  5
;3118          4647  5
;3119          4648  5                ! Advance to next entry.
;3120          4649  5                !
;3121          4650  5                Q = .Q + FID$C_LENGTH*2;
;3122          4651  4                END;
;3123          4652  3            END;
;3124          4653  3
;3125          4654  3
;3126          4655  3        ! Free the file ID block.
;3127          4656  3        !
;3128          4657  3        FREE_VM(REC_S_ENTRY, .P);
;3129          4658  2        END;
;3130          4659  2
;3131          4660  2
;3132          4661  2 ! Deassign the channel.
;3133          4662  2 !
;3134          4663  2 $DASSGN(CHAN=.INPUT_CHAN);
;3135          4664  2 INPUT_CHAN = 0;
;3136          4665  1 END;


                                                016F5          .BLKB   3
                                      0014  0008  016F8 P.AAH:  .WORD   8, 20                                          ;
                                        00000000' 016FC         .ADDRESS JPI_DATE                                      ;
                                        00000000  01700         .LONG   0                                              ;

                                                  OFFC 00000 POSTPROCESS_FILES:
                                                             .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11               ; 4474
                                      5E        B0   AE  9E 00002    MOVAB   -80(SP), SP                                ;
                                                     58  D4 00006    CLRL    IN_QUAL                                    ; 4516
                                                     5B  D4 00008    CLRL    WRITELOCKED                                ; 4517
                                            08   AE        40  8F 9A 0000A    MOVZBL  #64, FIB_DESC                      ; 4518
                                            0C   AE        10  AE 9E 0000F    MOVAB   FIB, FIB_DESC+4                    ; 4519
                                  08 00000000'   EF            02  E1 00014    BBC     #2, QUAL+14, 1$                    ; 4520
                                                     00000000G  8F  DD 0001C    PUSHL   #BACKUP$_STARTDELETE               ; 4521
                                                                06  11 00022    BRB     2$                                 ;
                                                     00000000G  8F  DD 00024 1$: PUSHL   #BACKUP$_STARTRECORD           ; 4522
                                        00000000G    00             01  FB 0002A 2$: CALLS   #1, LIB$SIGNAL              ;
                                                     50 00000000'   EF  D0 00031 3$: MOVL    INPUT_PROC_LIST, R0        ; 4527
                                                                03  12 00038    BNEQ    4$                                 ;
                                                              0196  31 0003A    BRW     22$                                ;
                                                     56             50  D0 0003D 4$: MOVL    R0, P                      ; 4532
                                        00000000'    EF             66  D0 00040    MOVL    (P), INPUT_PROC_LIST       ; 4533
                                                     58        04   A6  D1 00047    CMPL    4(P), IN_QUAL              ; 4538
                                                                44  13 0004B    BEQL    7$                                 ;
                                                     58        04   A6  D0 0004D    MOVL    4(P), IN_QUAL              ; 4544
                                                     50 00000000'   EF  D0 00051    MOVL    INPUT_CHAN, R0             ; 4549
                                                                09  13 00058    BEQL    5$                                 ;
                                                                50  DD 0005A    PUSHL   R0                                 ;
                                        00000000G    00             01  FB 0005C    CALLS   #1, SYS$DASSGN             ;
                                                                7E  7C 00063 5$: CLRQ    -(SP)                      ; 4550
```

```
                                 00000000'  EF  9F 00065          PUSHAB  INPUT_CHAN                                      ;
                                       10   A8  9F 0006B          PUSHAB  16(IN_QUAL)                                     ;
                          00000000G    00        04  FB 0006E    CALLS   #4, ASSIGN_INPUT_CHANNEL                        ;
                                       59        50  D0 00075    MOVL    R0, STATUS                                      ;
                                       14        59  E8 00078    BLBS    STATUS, 6$                                      ; 4551
                                                 59  DD 0007B    PUSHL   STATUS                                          ; 4555
                                           10    A8  9F 0007D    PUSHAB  16(IN_QUAL)                                     ; 4554
                                                 01  DD 00080    PUSHL   #1                                              ;
                                 00000000G       8F  DD 00082    PUSHL   #BACKUP$_OPENOUT+4                              ;
                          00000000G    00        04  FB 00088    CALLS   #4, LIB$SIGNAL                                  ;
                                                 5B  D4 0008F 6$: CLRL    WRITELOCKED                                     ; 4556
                                       03        5B  E9 00091 7$: BLBC    WRITELOCKED, 8$                                 ; 4562
                                               012B  31 00094    BRW     21$                                             ;
                                       57    0C  A6  9E 00097 8$: MOVAB   12(R6), Q                                       ; 4565
                                       5A    08  A6  3C 0009B    MOVZWL  8(P), I                                         ; 4566
                                                 5A  D6 0009F    INCL    I                                               ;
                                             0116  31 000A1    BRW     19$                                             ;
    0040    8F         00        6E        00  2C 000A4 9$: MOVC5   #0, (SP), #0, #64, FIB                           ; 4568
                                       10    AE      000AB
                                 14    AE        67  D0 000AD    MOVL    (Q), FIB+4                                      ; 4569
                                 18    AE    04  A7  B0 000B1    MOVW    4(Q), FIB+8                                     ; 4571
                          7B 00000000'  EF        02  E1 000B6    BBC     #2, QUAL+14, 12$                                ; 4574
                                       50    06  A7  9E 000BE    MOVAB   6(Q), R0                                        ; 4577
                                 1A    AE        60  D0 000C2    MOVL    (R0), FIB+10                                    ;
                                 1E    AE    04  A0  B0 000C6    MOVW    4(R0), FIB+14                                   ; 4579
                                 25    AE        08  88 000CB    BISB2   #8, FIB+21                                      ; 4580
                                                 7E  7C 000CF    CLRQ    -(SP)                                           ; 4589
                                                 7E  7C 000D1    CLRQ    -(SP)                                           ;
                                                 7E  D4 000D3    CLRL    -(SP)                                           ;
                                       1C    AE  9F 000D5    PUSHAB  FIB_DESC                                        ;
                                                 7E  7C 000D8    CLRQ    -(SP)                                           ;
                                       20    AE  9F 000DA    PUSHAB  IOSB                                            ;
                                    7E   0135    8F  3C 000DD    MOVZWL  #309, -(SP)                                     ;
                                      00000000' EF  DD 000E2    PUSHL   INPUT_CHAN                                      ;
                                                 7E  D4 000E8    CLRL    -(SP)                                           ;
                          00000000G    00        0C  FB 000EA    CALLS   #12, SYS$QIOW                                   ;
                                       59        50  D0 000F1    MOVL    R0, STATUS                                      ;
                                       06        59  E9 000F4    BLBC    STATUS, 10$                                     ; 4590
                                       59        6E  3C 000F7    MOVZWL  IOSB, STATUS                                    ;
                                       6C        59  E8 000FA    BLBS    STATUS, 13$                                     ; 4591
                                       52    10  A8  9E 000FD 10$: MOVAB   16(R8), R2                                     ; 4596
                                 025C  8F        59  B1 00101    CMPW    STATUS, #604                                    ; 4593
                                                 0C  12 00106    BNEQ    11$                                             ;
                                                 52  DD 00108    PUSHL   R2                                              ; 4596
                                                 01  DD 0010A    PUSHL   #1                                              ;
                                      00000000G  8F  DD 0010C    PUSHL   #BACKUP$_DELETESWL                              ;
                                                 6D  11 00112    BRB     15$                                             ;
                                                 59  DD 00114 11$: PUSHL   STATUS                                          ; 4608
                                       1C    AE  9A 00116    MOVZBL  FIB+8, -(SP)                                    ; 4607
                                       1E    AE  3C 0011A    MOVZWL  FIB+6, -(SP)                                    ; 4606
                                       20    AE  3C 0011E    MOVZWL  FIB+4, R0                                       ; 4605
                                       25    AE  9A 00122    MOVZBL  FIB+9, R1                                       ;
                    51                 51        10  78 00126    ASHL    #16, R1, R1                                     ;
                                             6140  9F 0012A    PUSHAB  (R1)[R0]                                        ;
                                                 52  DD 0012D    PUSHL   R2                                              ; 4604
                                                 04  DD 0012F    PUSHL   #4                                              ;
                                      00000000G  8F  DD 00131    PUSHL   #BACKUP$_DELETE                                 ;
```

```
                                  77  11 00137        BRB     17$                          ;
           12  AE                 20  88 00139 12$:   BISB2   #32, FIB+2                   ; 4616
                                  7E  D4 0013D        CLRL    -(SP)                        ; 4625
                        FEB1      CF  9F 0013F        PUSHAB  P.AAH                        ;
                                  7E  7C 00143        CLRQ    -(SP)                        ;
                                  7E  D4 00145        CLRL    -(SP)                        ;
                          1C      AE  9F 00147        PUSHAB  FIB_DESC                     ;
                                  7E  7C 0014A        CLRQ    -(SP)                        ;
                          20      AE  9F 0014C        PUSHAB  IOSB                         ;
                                  36  DD 0014F        PUSHL   #54                          ;
                   00000000'      EF  DD 00151        PUSHL   INPUT_CHAN                   ;
                                  7E  D4 00157        CLRL    -(SP)                        ;
      00000000G  00               0C  FB 00159        CALLS   #12, SYS$QIOW                ;
                 59               50  D0 00160        MOVL    R0, STATUS                   ;
                 06               59  E9 00163        BLBC    STATUS, 14$                  ; 4626
                 59               6E  3C 00166        MOVZWL  IOSB, STATUS                 ;
                 4B               59  E8 00169 13$:   BLBS    STATUS, 18$                  ; 4627
                 52       10      A8  9E 0016C 14$:   MOVAB   16(R8), R2                   ; 4632
           025C  8F               59  B1 00170        CMPW    STATUS, #604                 ; 4629
                                  16  12 00175        BNEQ    16$                          ;
                                  52  DD 00177        PUSHL   R2                           ; 4632
                                  01  DD 00179        PUSHL   #1                           ;
                   00000000G      8F  DD 0017B        PUSHL   #BACKUP$_RECORDSWL           ;
      00000000G  00               03  FB 00181 15$:   CALLS   #3, LIB$SIGNAL               ;
                 5B               01  D0 00188        MOVL    #1, WRITELOCKED              ; 4633
                                  35  11 0018B        BRB     21$                          ; 4631
                                  59  DD 0018D 16$:   PUSHL   STATUS                       ; 4644
                 7E       1C      AE  9A 0018F        MOVZBL  FIB+8, -(SP)                 ; 4643
                 7E       1E      AE  3C 00193        MOVZWL  FIB+6, -(SP)                 ; 4642
                 50       20      AE  3C 00197        MOVZWL  FIB+4, R0                    ; 4641
                 51       25      AE  9A 0019B        MOVZBL  FIB+9, R1                    ;
      51         51               10  78 0019F        ASHL    #16, R1, R1                  ;
                                6140  9F 001A3        PUSHAB  (R1)[R0]                     ;
                                  52  DD 001A6        PUSHL   R2                           ; 4640
                                  04  DD 001A8        PUSHL   #4                           ;
                   00000000G      8F  DD 001AA        PUSHL   #BACKUP$_WRITEBACK           ;
      00000000G  00               07  FB 001B0 17$:   CALLS   #7, LIB$SIGNAL               ;
                 57               0C  C0 001B7 18$:   ADDL2   #12, Q                       ; 4650
                 02               5A  F5 001BA 19$:   SOBGTR  I, 20$                       ; 4566
                                  03  11 001BD        BRB     21$                          ;
                                FEE2  31 001BF 20$:   BRW     9$                           ;
                                  56  DD 001C2 21$:   PUSHL   P                            ; 4657
                 7E     030C      8F  3C 001C4        MOVZWL  #780, -(SP)                  ;
      00000000G  00               02  FB 001C9        CALLS   #2, FREE_VM                  ;
                                FE5E  31 001D0        BRW     3$                           ; 4527
                   00000000'      EF  DD 001D3 22$:   PUSHL   INPUT_CHAN                   ; 4663
      00000000G  00               01  FB 001D9        CALLS   #1, SYS$DASSGN               ;
                   00000000'      EF  D4 001E0        CLRL    INPUT_CHAN                   ; 4664
                                  04 001E6            RET                                  ; 4665

; Routine Size:  487 bytes,    Routine Base:  CODE + 1704
```

```
;  3138          4666  1 %SBTTL 'SAVE_FILE_ID - add file ID list entry'
;  3139          4667  1 ROUTINE SAVE_FILE_ID: NOVALUE=
;  3140          4668  1
;  3141          4669  1 !++
;  3142          4670  1 !
;  3143          4671  1 ! FUNCTIONAL DESCRIPTION:
;  3144          4672  1 !       This routine adds an entry to the file ID list.
;  3145          4673  1 !
;  3146          4674  1 ! INPUT PARAMETERS:
;  3147          4675  1 !       NONE
;  3148          4676  1 !
;  3149          4677  1 ! IMPLICIT INPUTS:
;  3150          4678  1 !       INPUT_PROC_LIST - List head of list.
;  3151          4679  1 !       INPUT_NAM       - Contains file ID.
;  3152          4680  1 !
;  3153          4681  1 ! OUTPUT PARAMETERS:
;  3154          4682  1 !       NONE
;  3155          4683  1 !
;  3156          4684  1 ! IMPLICIT OUTPUTS:
;  3157          4685  1 !       NONE
;  3158          4686  1 !
;  3159          4687  1 ! ROUTINE VALUE:
;  3160          4688  1 !       NONE
;  3161          4689  1 !
;  3162          4690  1 ! SIDE EFFECTS:
;  3163          4691  1 !       File ID entered in structure.  New block may be allocated.
;  3164          4692  1 !
;  3165          4693  1 !--
;  3166          4694  1
;  3167          4695  2 BEGIN
;  3168          4696  2 LOCAL
;  3169          4697  2         P:              REF BBLOCK,     ! Pointer to block
;  3170          4698  2         Q:              REF BBLOCK;     ! Pointer to where FID is stored
;  3171          4699  2
;  3172          4700  2
;  3173          4701  2 P = .INPUT_PROC_LIST;
;  3174          4702  2 IF
;  3175          4703  3     BEGIN
;  3176          4704  3     IF .P EQL 0
;  3177          4705  3     THEN
;  3178          4706  3         TRUE
;  3179          4707  3     ELSE
;  3180          4708  3         .P[REC_USED] GEQU REC_MAX_COUNT OR
;  3181          4709  3         .P[REC_QUAL] NEQ .INPUT_QUAL OR
;  3182          4710  3         .P[REC_VOLUME] NEQ .RWSV_VOL_NUMBER
;  3183          4711  3     END
;  3184          4712  2 THEN
;  3185          4713  3     BEGIN
;  3186          4714  3     INPUT_PROC_LIST = GET_VM(REC_S_ENTRY);
;  3187          4715  3     INPUT_PROC_LIST[REC_NEXT] = .P;
;  3188          4716  3     INPUT_PROC_LIST[REC_QUAL] = .INPUT_QUAL;
;  3189          4717  3     INPUT_PROC_LIST[REC_USED] = 0;
;  3190          4718  3     INPUT_PROC_LIST[REC_VOLUME] = .RWSV_VOL_NUMBER;
;  3191          4719  3     P = .INPUT_PROC_LIST;
;  3192          4720  2     END;
;  3193          4721  2
;  3194          4722  2
```

```
;  3195          4723  2 Q = P[REC_FID_BASE] + FID$C_LENGTH*2 * .P[REC_USED];
;  3196          4724  2 BBLOCK[Q[REC_FID], FID$W_NUM] = .INPUT_NAM[NAM$W_FID_NUM];
;  3197          4725  2 BBLOCK[Q[REC_FID], FID$W_SEQ] = .INPUT_NAM[NAM$W_FID_SEQ];
;  3198          4726  2 BBLOCK[Q[REC_FID], FID$W_RVN] = .INPUT_NAM[NAM$W_FID_RVN];
;  3199          4727  2 BBLOCK[Q[REC_DID], FID$W_NUM] = .INPUT_NAM[NAM$W_DID_NUM];
;  3200          4728  2 BBLOCK[Q[REC_DID], FID$W_SEQ] = .INPUT_NAM[NAM$W_DID_SEQ];
;  3201          4729  2 BBLOCK[Q[REC_DID], FID$W_RVN] = .INPUT_NAM[NAM$W_DID_RVN];
;  3202          4730  2 P[REC_USED] = .P[REC_USED] + 1;
;  3203          4731  1 END;


                                000C 00000 SAVE_FILE_ID:
                                           .WORD   Save R2,R3                                  ; 4667
                    53 00000000'   EF  9E 00002    MOVAB   INPUT_PROC_LIST, R3
                    52             63  D0 00009    MOVL    INPUT_PROC_LIST, P                  ; 4701
                                   15  13 0000C    BEQL    1$                                  ; 4704
                    3F       08    A2  B1 0000E    CMPW    8(P), #63                           ; 4708
                                   0F  1A 00012    BGTRU   1$
                 94 A3       04    A2  D1 00014    CMPL    4(P), INPUT_QUAL                    ; 4709
                                   08  12 00019    BNEQ    1$
              FE5C C3        0A    A2  B1 0001B    CMPW    10(P), RWSV_VOL_NUMBER              ; 4710
                                   23  13 00021    BEQL    2$
                    7E       030C  8F  3C 00023 1$: MOVZWL  #780, -(SP)                        ; 4714
          00000000G 00             01  FB 00028    CALLS   #1, GET_VM
                    63             50  D0 0002F    MOVL    R0, INPUT_PROC_LIST
                    60             52  D0 00032    MOVL    P, (R0)                             ; 4715
                 04 A0       94    A3  D0 00035    MOVL    INPUT_QUAL, 4(R0)                   ; 4716
                                08  A0  B4 0003A    CLRW    8(R0)                               ; 4717
                 0A A0       FE5C  C3  B0 0003D    MOVW    RWSV_VOL_NUMBER, 10(R0)             ; 4718
                    52             50  D0 00043    MOVL    R0, P                               ; 4719
                    50       08    A2  3C 00046 2$: MOVZWL  8(P), R0                          ; 4723
                    50             0C  C4 0004A    MULL2   #12, R0
                    51       0C A042  9E 0004D    MOVAB   12(R0)[P], Q
                    50       8C    A3  D0 00052    MOVL    INPUT_NAM, R0                       ; 4724
                    81       24    A0  D0 00056    MOVL    36(R0), (Q)+
                    81       28    A0  B0 0005A    MOVW    40(R0), (Q)+                        ; 4726
                    61       2A    A0  D0 0005E    MOVL    42(R0), (R1)                        ; 4727
                 04 A1       2E    A0  BC 00062    MOVW    46(R0), 4(R1)                       ; 4729
                                08  A2  B6 00067    INCW    8(P)                                ; 4730
                                       04 0006A    RET                                         ; 4731

; Routine Size:  107 bytes,    Routine Base:  CODE + 18EB
```

```
;  3205     4732  1 %SBTTL 'SAVE_ONE_FILE_HANDLER - condition handler for SAVE_ONE_FILE'
;  3206     4733  1 ROUTINE SAVE_ONE_FILE_HANDLER(SIG,MECH)=
;  3207     4734  1
;  3208     4735  1 !++
;  3209     4736  1 !
;  3210     4737  1 ! FUNCTIONAL DESCRIPTION:
;  3211     4738  1 !       This routine is a condition handler for routine SAVE_ONE_FILE.  It
;  3212     4739  1 !       takes care of unwinding the stack when a failure to continue a save on
;  3213     4740  1 !       a continuation tape occurs, and of recording the fact that an error
;  3214     4741  1 !       occurred during file processing.
;  3215     4742  1 !
;  3216     4743  1 ! INPUT PARAMETERS:
;  3217     4744  1 !       SIG             - Standard VMS handler parameters
;  3218     4745  1 !       MECH            -
;  3219     4746  1 !
;  3220     4747  1 ! IMPLICIT INPUTS:
;  3221     4748  1 !       NONE
;  3222     4749  1 !
;  3223     4750  1 ! OUTPUT PARAMETERS:
;  3224     4751  1 !       NONE
;  3225     4752  1 !
;  3226     4753  1 ! IMPLICIT OUTPUTS:
;  3227     4754  1 !       NONE
;  3228     4755  1 !
;  3229     4756  1 ! ROUTINE VALUE:
;  3230     4757  1 !       SS$_RESIGNAL
;  3231     4758  1 !
;  3232     4759  1 ! SIDE EFFECTS:
;  3233     4760  1 !       NONE
;  3234     4761  1 !
;  3235     4762  1 !--
;  3236     4763  1
;  3237     4764  2 BEGIN
;  3238     4765  2 MAP
;  3239     4766  2         SIG:            REF BBLOCK,     ! Signal parameters
;  3240     4767  2         MECH:           REF BBLOCK;     ! Mechanism parameters
;  3241     4768  2
;  3242     4769  2
;  3243     4770  2 IF .SIG[CHF$L_SIG_NAME] EQL BACKUP$_CONTINUE
;  3244     4771  2 THEN
;  3245     4772  3     BEGIN
;  3246     4773  3     INPUT_FLAGS[EOV_SAVING] = FALSE;
;  3247     4774  3     SIG[CHF$L_SIG_ARGS] = .SIG[CHF$L_SIG_ARGS] - 2;
;  3248     4775  3     MECH[CHF$[_MCA_SAVR0] = TRUE;
;  3249     4776  3     $PUTMSG(MSGVEC=.SIG, ACTRTN=PUTMSG_ACTRTN);
;  3250     4777  3     $UNWIND();
;  3251     4778  3     END
;  3252     4779  3
;  3253     4780  3
;  3254     4781  2 ELSE IF NOT .SIG[CHF$L_SIG_NAME]
;  3255     4782  2 THEN
;  3256     4783  2     INPUT_FLAGS[INPUT_SAVE_OK] = FALSE;
;  3257     4784  2
;  3258     4785  2
;  3259     4786  2 SS$_RESIGNAL
;  3260     4787  1 END;
```

```
                                                        .EXTRN  SYS$PUTMSG

                                              0004 00000 SAVE_ONE_FILE_HANDLER:
                                                        .WORD   Save R2                                 ; 4733
                    52 00000000'   EF  9E C0002         MOVAB   INPUT_FLAGS, R2                         ;
                    51         04  AC  D0 00009         MOVL    SIG, R1                                 ; 4770
00000000G           8F         04  A1  D1 0000D         CMPL    4(R1), #BACKUP$_CONTINUE                ;
                                   2A  12 00015         BNEQ    1$                                      ;
                    62             04  8A 00017         BICB2   #4, INPUT_FLAGS                         ; 4773
                    61             02  C2 0001A         SUBL2   #2, (R1)                                ; 4774
                    50         08  AC  D0 0001D         MOVL    MECH, R0                                ; 4775
                 0C AO             01  D0 00021         MOVL    #1, 12(R0)                              ;
                                   7E  7C 00025         CLRQ    -(SP)                                   ; 4776
                       00000000G   00  9F 00027         PUSHAB  PUTMSG_ACTRTN                           ;
                                   51  DD 0002D         PUSHL   R1                                      ;
00000000G           00             04  FB 0002F         CALLS   #4, SYS$PUTMSG                          ;
                                   7E  7C 00036         CLRQ    -(SP)                                   ; 4777
00000000G           00             02  FB 00038         CALLS   #2, SYS$UNWIND                          ;
                                   07  11 0003F         BRB     2$                                      ; 4770
                    03         04  A1  E8 00041 1$:     BLBS    4(R1), 2$                               ; 4781
                    62             20  8A 00045         BICB2   #32, INPUT_FLAGS                        ; 4783
                    50       0918  8F  3C 00048 2$:     MOVZWL  #2328, R0                               ; 4787
                                   04  0004D             RET                                             ;

; Routine Size:  78 bytes,    Routine Base:  CODE + 1956
```

```
;  3262          4788  1 %SBTTL 'SAVE_ONE_FILE - save an input file'
;  3263          4789  1 GLOBAL ROUTINE SAVE_ONE_FILE=
;  3264          4790  1
;  3265          4791  1 !++
;  3266          4792  1 !
;  3267          4793  1 ! FUNCTIONAL DESCRIPTION:
;  3268          4794  1 !       This is the success routine for input file processing.
;  3269          4795  1 !
;  3270          4796  1 ! INPUT PARAMETERS:
;  3271          4797  1 !       NONE
;  3272          4798  1 !
;  3273          4799  1 ! IMPLICIT INPUTS:
;  3274          4800  1 !       INPUT_FAB       - Pointer to the FAB.
;  3275          4801  1 !       INPUT_NAM       - Pointer to the NAM block.
;  3276          4802  1 !
;  3277          4803  1 ! OUTPUT PARAMETERS:
;  3278          4804  1 !       NONE
;  3279          4805  1 !
;  3280          4806  1 ! IMPLICIT OUTPUTS:
;  3281          4807  1 !       NONE
;  3282          4808  1 !
;  3283          4809  1 ! ROUTINE VALUE:
;  3284          4810  1 !       True if the file was accessed without incurring an SS$_NOSUCHFILE
;  3285          4811  1 !       error; false otherwise.
;  3286          4812  1 !
;  3287          4813  1 ! SIDE EFFECTS:
;  3288          4814  1 !       NONE
;  3289          4815  1 !
;  3290          4816  1 !--
;  3291          4817  1
;  3292          4818  2 BEGIN
;  3293          4819  2 LOCAL
;  3294          4820  2         FIB:            BBLOCK[FIB$C_LENGTH],   ! FIB
;  3295          4821  2         FIB_DESC:       VECTOR[2],      ! Descriptor for FIB
;  3296          4822  2         INPUT_HDR:      BBLOCK[512],    ! Buffer for file header
;  3297          4823  2         FILE_ACL        : BBLOCK [ATR$S_READACL],       ! Storage for small ACL
;  3298          4824  2         ATR_DESC:       BBLOCK[36],     ! ACP attributes list
;  3299          4825  2         STATBLK:        BBLOCK[SBK$C_LENGTH],   ! Statistics block
;  3300          4826  2         BLOCKS,                         ! Blocks of file to be copied
;  3301          4827  2         ALWAYS_NOBACK:  BITVECTOR[32],  ! Unconditionally NOBACKUP files
;  3302          4828  2         NOBACK_FLAG:    BYTE,           ! Flag signifing that data was not backed up
;  3303          4829  2         STATUS,                         ! Status return
;  3304          4830  2         IOSB:           VECTOR[4,WORD]; ! I/O status block
;  3305          4831  2 BUILTIN
;  3306          4832  2         FP;
;  3307          4833  2
;  3308          4834  2
;  3309          4835  2 ! Establish the handler.
;  3310          4836  2 !
;  3311          4837  2 .FP = SAVE_ONE_FILE_HANDLER;
;  3312          4838  2
;  3313          4839  2
;  3314          4840  2 ! Initialize the filename descriptor and the FIB.
;  3315          4841  2 !
;  3316          4842  2 CH$FILL (0, FIB$C_LENGTH, FIB);
;  3317          4843  2 EXTRACT_DIR_FILENAME(.INPUT_FAB, INPUT_NAMEDESC);
;  3318          4844  2 FIB[FIB$L_ACCTL] = FIB$M_NOWRITE;
```

```
;  3319     4845  2 INPUT_FLAGS[INPUT_IGNO_INTE] = FALSE;
;  3320     4846  2 IF
;  3321     4847  2     .QUAL[QUAL_IGNO_INTE] OR
;  3322     4848  3     (.INPUT_NAMEDESC[DSC$W_LENGTH] EQL %CHARCOUNT('[000000]QUOTA.SYS;1') AND
;  3323     4849  3     CH$EQL(
;  3324     4850  3         %CHARCOUNT('[000000]QUOTA.SYS;1'),
;  3325     4851  3         UPLIT BYTE('[000000]QUOTA.SYS;1'),
;  3326     4852  3         %CHARCOUNT('[000000]QUOTA.SYS;1'),
;  3327     4853  3         .INPUT_NAMEDESC[DSC$A_POINTER]))
;  3328     4854  2 THEN
;  3329     4855  3     BEGIN
;  3330     4856  3     INPUT_FLAGS[INPUT_IGNO_INTE] = TRUE;
;  3331     4857  3     FIB[FIB$L_ACCTL] = FIB$M_NOLOCK;
;  3332     4858  2     END;
;  3333     4859  2 FIB[FIB$V_NORECORD] = TRUE;
;  3334     4860  2 FIB[FIB$W_FID_NUM] = .INPUT_NAM[NAM$W_FID_NUM];
;  3335     4861  2 FIB[FIB$W_FID_SEQ] = .INPUT_NAM[NAM$W_FID_SEQ];
;  3336     4862  2 FIB[FIB$W_FID_RVN] = .INPUT_NAM[NAM$W_FID_RVN];
;  3337     4863  2 FIB_DESC[0] = FIB$C_LENGTH;
;  3338     4864  2 FIB_DESC[1] = FIB;
;  3339     4865  2
;  3340     4866  2
;  3341     4867  2 ! Initialize the ACP attributes list.
;  3342     4868  2 !
;  3343     4869  2 ATR_DESC[0,0,16,0] = ATR$S_HEADER;
;  3344     4870  2 ATR_DESC[2,0,16,0] = ATR$C_HEADER;
;  3345     4871  2 ATR_DESC[4,0,32,0] = INPUT_HDR;
;  3346     4872  2 ATR_DESC[8,0,16,0] = SBK$C_LENGTH;
;  3347     4873  2 ATR_DESC[10,0,16,0] = ATR$C_STATBLK;
;  3348     4874  2 ATR_DESC[12,0,32,0] = STATBLK;
;  3349     4875  2 ATR_DESC[16,0,16,0] = ATR$S_READACL;
;  3350     4876  2 ATR_DESC[18,0,16,0] = ATR$C_READACL;
;  3351     4877  2 ATR_DESC[20,0,32,0] = FILE_ACL;
;  3352     4878  2 ATR_DESC[24,0,16,0] = ATR$S_ACLLENGTH;
;  3353     4879  2 ATR_DESC[26,0,16,0] = ATR$C_ACLLENGTH;
;  3354     4880  2 ATR_DESC[28,0,32,0] = ACL_LENGTH;
;  3355     4881  2 ATR_DESC[32,0,32,0] = 0;
;  3356     4882  2
;  3357     4883  2
;  3358     4884  2 ! Access the file and read the file header.
;  3359     4885  2 !
;  3360   P 4886  2 STATUS = $QIOW(
;  3361   P 4887  2     FUNC=IO$_ACCESS OR IO$M_ACCESS,
;  3362   P 4888  2     CHAN=.INPUT_CHAN,
;  3363   P 4889  2     IOSB=IOSB,
;  3364   P 4890  2     P1=FIB_DESC,
;  3365     4891  2     P5=ATR_DESC);
;  3366     4892  2 IF .STATUS THEN STATUS = .IOSB[0];
;  3367     4893  2 IF NOT .STATUS
;  3368     4894  2 THEN
;  3369     4895  3     BEGIN
;  3370     4896  3     FILE_ERROR(
;  3371     4897  3         BACKUP$_OPENIN + STS$K_ERROR,
;  3372     4898  3         .INPUT_FAB,
;  3373     4899  3         .STATUS);
;  3374     4900  3     IF .STATUS<0,16> EQL SS$_NOSUCHFILE THEN RETURN FALSE;
;  3375     4901  3     RETURN TRUE;
```

```
;  3376          4902  2      END;
;  3377          4903  2  INPUT_FLAGS[INPUT_OPEN] = TRUE;
;  3378          4904  2  IF .ACL_LENGTH LEQU ATR$S_READACL THEN ACL_BUFFER = FILE_ACL;
;  3379          4905  2
;  3380          4906  2
;  3381          4907  2  ! Check for file opened under interlock.
;  3382          4908  2  !
;  3383          4909  2  IF .INPUT_FLAGS[INPUT_IGNO_INTE] AND .STATBLK[SBK$W_WCNT] NEQ 0
;  3384          4910  2  THEN
;  3385          4911  2      FILE_ERROR(BACKUP$_ACCONFLICT, .INPUT_FAB);
;  3386          4912  2
;  3387          4913  2
;  3388          4914  2  ! Put the attributes in normal format.
;  3389          4915  2  !
;  3390          4916  2  INPUT_STATBLK[SBK$L_STLBN] = ROT(.STATBLK[SBK$L_STLBN], 16);
;  3391          4917  2  INPUT_STATBLK[SBK$L_FILESIZE] = ROT(.STATBLK[SBK$L_FILESIZE], 16);
;  3392          4918  2
;  3393          4919  2
;  3394          4920  2  ! Initialize the structure-independent attributes area.
;  3395          4921  2  !
;  3396          4922  2  INIT_ATTR(INPUT_HDR);
;  3397          4923  2
;  3398          4924  2
;  3399          4925  2  ! Establish the number of blocks to be copied.  Copy up to the EOF block,
;  3400          4926  2  ! except for other than sequential organization.
;  3401          4927  2  !
;  3402          4928  2  INPUT_BLOCK = 1;
;  3403          4929  2  INPUT_MAXBLOCK = ROT(.INPUT_RECATTR[FAT$L_EFBLK], 16);
;  3404          4930  2  IF .INPUT_RECATTR[FAT$W_FFBYTE] EQL 0
;  3405          4931  2      THEN INPUT_MAXBLOCK = .INPUT_MAXBLOCK - 1;
;  3406          4932  2  IF
;  3407          4933  2      .INPUT_MAXBLOCK GTRU .INPUT_STATBLK[SBK$L_FILESIZE] OR
;  3408          4934  2      .INPUT_RECATTR[FAT$V_FILEORG] NEQ FAT$C_SEQUENTIAL
;  3409          4935  2  THEN
;  3410          4936  2      INPUT_MAXBLOCK = .INPUT_STATBLK[SBK$L_FILESIZE];
;  3411          4937  2
;  3412          4938  2
;  3413          4939  2  ! Get the file placement data.
;  3414          4940  2  !
;  3415          4941  2  IF .QUAL[QUAL_IMAG] THEN GET_PLC_DATA(INPUT_HDR);
;  3416          4942  2
;  3417          4943  2
;  3418          4944  2  ! Evaluate the file selection criteria.
;  3419          4945  2  ! If they fail, close the file and return.
;  3420          4946  2  !
;  3421          4947  2  IF
;  3422          4948  3      BEGIN
;  3423          4949  3      IF .QUAL[QUAL_IMAG]
;  3424          4950  3      THEN
;  3425          4951  3          IF .QUAL[QUAL_VOLU]
;  3426          4952  3          THEN
;  3427          4953  3              NOT .INPUT_FLAGS[INPUT_ON_RVN]
;  3428          4954  3          ELSE
;  3429          4955  3              FALSE
;  3430          4956  3      ELSE
;  3431          4957  3          IF .QUAL[QUAL_FAST]
;  3432          4958  3          THEN
```

```
;3433        4959  3              NOT SELECT_INPUT_FILE(%B'100')
;3434        4960  3          ELSE
;3435        4961  3              IF NOT SELECT_INPUT_FILE(%B'001')
;3436        4962  3              THEN
;3437        4963  3                  TRUE
;3438        4964  3              ELSE
;3439        4965  3                  NOT SELECT_INPUT_FILE(%B'100')
;3440        4966  3      END
;3441        4967  2  THEN
;3442        4968  3      BEGIN
;3443      P 4969  3      STATUS = $QIOW(
;3444      P 4970  3          FUNC=IO$_DEACCESS,
;3445      P 4971  3          CHAN=.INPUT_CHAN,
;3446        4972  3          IOSB=IOSB);
;3447        4973  3      IF .STATUS THEN STATUS = .IOSB[0];
;3448        4974  3      IF NOT .STATUS
;3449        4975  3      THEN
;3450        4976  3          FILE_ERROR(
;3451        4977  3              BACKUP$_CLOSEIN + STS$K_ERROR,
;3452        4978  3              .INPUT_FAB,
;3453        4979  3              .STATUS);
;3454        4980  3      INPUT_FLAGS[INPUT_OPEN] = FALSE;
;3455        4981  3      RETURN TRUE;
;3456        4982  2      END;
;3457        4983  2
;3458        4984  2
;3459        4985  2  ! Note that a file was found.
;3460        4986  2  ! Increment the count of uses of this input filespec.
;3461        4987  2  !
;3462        4988  2  IF
;3463        4989  2      NOT .DIR_STATUS[D_STAT_VALID] OR
;3464        4990  2      .DIR_STATUS[D_STAT_DIR_SEL]
;3465        4991  2  THEN
;3466        4992  2      COM_FLAGS[COM_FILESEEN] = TRUE;
;3467        4993  2
;3468        4994  2  INPUT_QUAL[QUAL_USE_COUNT] = .INPUT_QUAL[QUAL_USE_COUNT] + 1;
;3469        4995  2  INPUT_FLAGS[INPUT_SAVE_OK] = TRUE;
;3470        4996  2
;3471        4997  2
;3472        4998  2  ! Generate the file attribute record.
;3473        4999  2  !
;3474        5000  2  FILE_ATTRIBUTES(INPUT_HDR);
;3475        5001  2
;3476        5002  2
;3477        5003  2  ! A file with the NOBACKUP attribute has its data copied only if the
;3478        5004  2  ! IGNORE=NOBACKUP qualifier is in effect.  Certain of the reserved files
;3479        5005  2  ! are ALWAYS implicitly NOBACKUP.  If warranted, copy the blocks.
;3480        5006  2  !
;3481        5007  2  ALWAYS_NOBACK = 1 ^ FID$C_INDEXF OR 1 ^ FID$C_BITMAP OR 1 ^ FID$C_BADBLK;
;3482        5008  2  IF .INPUT_HDR[FH2$B_STRUC[EV] EQL 2 THEN ALWAYS_NOBACK[FID$C_BADLOG] = TRUE;
;3483        5009  2  IF
;3484        5010  3      (NOT .INPUT_FILECHAR[FCH$V_NOBACKUP] OR
;3485        5011  3      .QUAL[QUAL_IGNO_NOBA] OR
;3486        5012  3      .INPUT_FILECHAR[FCH$V_DIRECTORY])
;3487        5013  3  AND NOT (
;3488        5014  3      .INPUT_NAM[NAM$W_FID_NUM] LEQU 15 AND
;3489        5015  3      .INPUT_NAM[NAM$B_FID_NMX] EQL 0 AND
```

```
;  3490          5016  3      .ALWAYS_NOBACK[.INPUT_NAM[NAM$W_FID_NUM]])
;  3491          5017  2  THEN
;  3492          5018  2      NOBACK_FLAG = FALSE
;  3493          5019  2  ELSE
;  3494          5020  2      NOBACK_FLAG = TRUE ;
;  3495          5021  2
;  3496          5022  2  ! Generate the journal record.
;  3497          5023  2  !
;  3498          5024  2  IF
;  3499          5025  2      .QUAL[QUAL_JOUR] AND
;  3500          5026  2      NOT .INPUT_FILECHAR[FCH$V_DIRECTORY]
;  3501          5027  2  THEN
;  3502          5028  2      WRITE_JOUR_FILE(.NOBACK_FLAG);
;  3503          5029  2
;  3504          5030  2
;  3505          5031  2  IF .NOBACK_FLAG
;  3506          5032  2  THEN
;  3507          5033  3      BEGIN
;  3508          5034  3      ! Inform the user that the file data was not saved
;  3509          5035  3      ! unless this is one of the special cases in which
;  3510          5036  3      ! case let us not trouble the user with the noise.
;  3511          5037  3      !
;  3512          5038  4      IF NOT (
;  3513          5039  4      .INPUT_NAM[NAM$W_FID_NUM] LEQU 15 AND
;  3514          5040  4      .INPUT_NAM[NAM$B_FID_NMX] EQL 0 AND
;  3515          5041  4      .ALWAYS_NOBACK[.INPUT_NAM[NAM$W_FID_NUM]])
;  3516          5042  3          THEN FILE_ERROR ( BACKUP$_NOBACKUP, .INPUT_FAB ) ;
;  3517          5043  3      END
;  3518          5044  2  ELSE
;  3519          5045  3      BEGIN
;  3520          5046  3      ! Copy the file data
;  3521          5047  3      !
;  3522          5048  3      NOBACK_FLAG = FALSE ;
;  3523          5049  3      IF .QUAL[QUAL_VOLU]
;  3524          5050  3      THEN
;  3525          5051  4          BEGIN
;  3526          5052  4          LOCAL
;  3527          5053  4              VBN:        REF BBLOCK;
;  3528          5054  4
;  3529          5055  4          VBN = .INPUT_VBN_LIST[0];
;  3530          5056  4          WHILE .VBN NEQ INPUT_VBN_LIST[0] DO
;  3531          5057  5              BEGIN
;  3532          5058  5              INPUT_BLOCK = .VBN[VBN_FIRST];
;  3533          5059  5              INPUT_MAXBLOCK = .VBN[VBN_LAST];
;  3534          5060  5              UNTIL SAVE_BLOCKS() DO 0;
;  3535          5061  5              VBN = .VBN[PLC_FLINK];
;  3536          5062  4              END;
;  3537          5063  4          END
;  3538          5064  3      ELSE
;  3539          5065  3          UNTIL SAVE_BLOCKS() DO 0;
;  3540          5066  2      END;
;  3541          5067  2
;  3542          5068  2
;  3543          5069  2  ! If doing a copy operation, close the output file (and verify it if this
;  3544          5070  2  ! has been requested.)
;  3545          5071  2  !
;  3546          5072  2  IF .QUAL[QUAL_OF11]
```

```
;  3547          5073  2 THEN
;  3548          5074  3     BEGIN
;  3549          5075  3     SAVE_WRITE();
;  3550          5076  3     CLOSE_RESTORE();
;  3551          5077  2     END;
;  3552          5078  2
;  3553          5079  2
;  3554          5080  2 ! Free the placement data.
;  3555          5081  2 !
;  3556          5082  2 IF .QUAL[QUAL_IMAG] THEN FREE_PLC_BLOCKS();
;  3557          5083  2
;  3558          5084  2
;  3559          5085  2 ! Deaccess the file.
;  3560          5086  2 !
;  3561        P 5087  2 STATUS = $QIOW(
;  3562        P 5088  2     FUNC=IO$_DEACCESS,
;  3563        P 5089  2     CHAN=.INPUT_CHAN,
;  3564          5090  2     IOSB=IOSB);
;  3565          5091  2 IF .STATUS THEN STATUS = .IOSB[0];
;  3566          5092  2 IF NOT .STATUS
;  3567          5093  2 THEN
;  3568          5094  2     FILE_ERROR(
;  3569          5095  2         BACKUP$_CLOSEIN + STS$K_ERROR,
;  3570          5096  2         .INPUT_FAB,
;  3571          5097  2         .STATUS);
;  3572          5098  2 INPUT_FLAGS[INPUT_OPEN] = FALSE;
;  3573          5099  2
;  3574          5100  2
;  3575          5101  2 ! If the LOG qualifier is in effect, log the file.
;  3576          5102  2 !
;  3577          5103  2 IF .QUAL[QUAL_LOG] AND NOT .QUAL[QUAL_OF11]
;  3578          5104  2 THEN
;  3579          5105  2     IF .NOBACK_FLAG
;  3580          5106  2     THEN
;  3581          5107  2         FILE_ERROR(
;  3582          5108  2             BACKUP$_HEADCOPIED,
;  3583          5109  2             .INPUT_FAB)
;  3584          5110  2     ELSE
;  3585          5111  2         FILE_ERROR(
;  3586          5112  2             BACKUP$_COPIED,
;  3587          5113  2             .INPUT_FAB);
;  3588          5114  2
;  3589          5115  2
;  3590          5116  2 ! Record that the file was successfully processed if appropriate.  Note that
;  3591          5117  2 ! /RECORD applies to a directory only if it was explicitly selected and that
;  3592          5118  2 ! /DELETE never applies to a directory.
;  3593          5119  2 !
;  3594          5120  2 IF
;  3595          5121  3     (.QUAL[QUAL_RECO] AND (.DIR_STATUS EQL 0 OR .DIR_STATUS[D_STAT_DIR_SEL]))
;  3596          5122  2 OR
;  3597          5123  3     (.QUAL[QUAL_DELE] AND .DIR_STATUS EQL 0)
;  3598          5124  2 THEN
;  3599          5125  2     IF .INPUT_FLAGS[INPUT_SAVE_OK]
;  3600          5126  2     THEN
;  3601          5127  2         SAVE_FILE_ID();
;  3602          5128  2
;  3603          5129  2
```

```
: 3604          5130  2 TRUE
: 3605          5131  1 END;


53  2E  41  54  4F  55  51  5D  30  30  30  30  30  30  5B  019A4 P.AAI:  .ASCII  \[000000]QUOTA.SYS;1\                    ;
                                        31  3B  53  59  019B3                                                            ;


                                                    07FC 00000          .ENTRY  SAVE_ONE_FILE, Save R2,R3,R4,R5,R6,R7,R8,-  ; 4789
                                                                                R9,R10                                      ;
                                    5A 00000000G    8F  D0 00002        MOVL    #BACKUP$_CLOSEIN+2, R10                     ;
                                    59 00000000G    00  9E 00009        MOVAB   SYS$QIOW, R9                                ;
                                    58 00000000G    00  9E 00010        MOVAB   FILE_ERROR, R8                              ;
                                    57      FBC2    CF  9E 00017        MOVAB   SELECT_INPUT_FILE, R7                       ;
                                    56 00000000'    EF  9E 0001C        MOVAB   INPUT_FLAGS, R6                             ;
                                    5E      FB6C    CE  9E 00023        MOVAB   -1172(SP), SP                               ;
                                    6D      03C2    C7  9E 00028        MOVAB   SAVE_ONE_FILE_HANDLER, (FP)                 ; 4837
    0040    8F          00          6E              00  2C 0002D        MOVC5   #0, (SP), #0, #64, FIB                      ; 4842
                                                C0  AD     00034                                                            ;
                                                24  A6  9F 00036        PUSHAB  INPUT_NAMEDESC                              ; 4843
                                                04  A6  DD 00039        PUSHL   INPUT_FAB                                   ;
                        00000000G   00              02  FB 0003C        CALLS   #2, EXTRACT_DIR_FILENAME                    ;
                              C0    AD              01  D0 00043        MOVL    #1, FIB                                     ; 4844
                                    66              10  8A 00047        BICB2   #16, INPUT_FLAGS                            ; 4845
                        0E      FF42    C6          02  E0 0004A        BBS     #2, QUAL+10, 1$                             ; 4847
                                        13      24  A6  B1 00050        CMPW    INPUT_NAMEDESC, #19                         ; 4848
                                                13  12 00054        BNEQ    2$                                          ;
                    28  B6          93  AF              13  29 00056        CMPC3   #19, P.AAI, @INPUT_NAMEDESC+4               ; 4849
                                                0B  12 0005C        BNEQ    2$                                          ;
                                    66              10  88 0005E 1$:     BISB2   #16, INPUT_FLAGS                            ; 4856
                              C0    AD  00100000    8F  D0 00061        MOVL    #1048576, FIB                               ; 4857
                              C2    AD              20  88 00069 2$:     BISB2   #32, FIB+2                                  ; 4859
                                    50          08  A6  D0 0006D        MOVL    INPUT_NAM, R0                               ; 4860
                              C4    AD          24  A0  D0 00071        MOVL    36(R0), FIB+4                               ;
                              C8    AD          28  A0  B0 00076        MOVW    40(R0), FIB+8                               ; 4862
                              B8    AD          40  8F  9A 0007B        MOVZBL  #64, FIB_DESC                               ; 4863
                              BC    AD          C0  AD  9E 00080        MOVAB   FIB, FIB_DESC+4                             ; 4864
                              28    AE  000A0200    8F  D0 00085        MOVL    #655872, ATR_DESC                           ; 4869
                              2C    AE      FDB8    CD  9E 0008D        MOVAB   INPUT_HDR, ATR_DESC+4                       ; 4871
                              30    AE  00090020    8F  D0 00093        MOVL    #589856, ATR_DESC+8                         ; 4872
                              34    AE          08  AE  9E 0009B        MOVAB   STATBLK, ATR_DESC+12                        ; 4874
                              38    AE  00250200    8F  D0 000A0        MOVL    #2425344, ATR_DESC+16                       ; 4875
                              3C    AE          4C  AE  9E 000A8        MOVAB   FILE_ACL, ATR_DESC+20                       ; 4877
                              40    AE  00260004    8F  D0 000AD        MOVL    #2490372, ATR_DESC+24                       ; 4878
                              44    AE      0604    C6  9E 000B5        MOVAB   ACL_LENGTH, ATR_DESC+28                     ; 4880
                                            48  AE  D4 000BB        CLRL    ATR_DESC+32                                 ; 4881
                                                7E  D4 000BE        CLRL    -(SP)                                       ; 4891
                                            2C  AE  9F 000C0        PUSHAB  ATR_DESC                                    ;
                                                7E  7C 000C3        CLRQ    -(SP)                                       ;
                                                7E  D4 000C5        CLRL    -(SP)                                       ;
                                            B8  AD  9F 000C7        PUSHAB  FIB_DESC                                    ;
                                                7E  7C 000CA        CLRQ    -(SP)                                       ;
                                            20  AE  9F 000CC        PUSHAB  IOSB                                        ;
                                    7E      72  8F  9A 000CF        MOVZBL  #114, -(SP)                                 ;
                                            FC  A6  DD 000D3        PUSHL   INPUT_CHAN                                  ;
```

```
                                                7E  D4 000D6          CLRL    -(SP)
                                             69 0C  FB 000D8          CALLS   #12, SYS$QIOW                                   ;
                                             53 50  D0 000DB          MOVL    R0, STATUS                                      ;
                                             06 53  E9 000DE          BLBC    STATUS, 3$                                      ; 4892
                                             53 6E  3C 000E1          MOVZWL  IOSB, STATUS                                    ;
                                             1B 53  E8 000E4          BLBS    STATUS, 5$                                      ; 4893
                                                53  DD 000E7 3$:      PUSHL   STATUS                                          ; 4899
                                          04    A6  DD 000E9          PUSHL   INPUT_FAB                                       ; 4898
                                    00000000G   8F  DD 000EC          PUSHL   #BACKUP$_OPENIN+2                               ; 4897
                                             68 03  FB 000F2          CALLS   #3, FILE_ERROR                                  ;
                                       0910  8F  53  B1 000F5          CMPW    STATUS, #2320                                   ; 4900
                                             03  13 000FA          BEQL    4$                                              ;
                                           023B  31 000FC          BRW     40$                                             ;
                                             50  D4 000FF 4$:      CLRL    R0                                              ;
                                                 04 00101          RET                                                     ;
                                             66  01  88 00102 5$:      BISB2   #1, INPUT_FLAGS                                 ; 4903
                                 00000200    8F      0604  C6  D1 00105          CMPL    ACL_LENGTH, #512                        ; 4904
                                             06  1A 0010E          BGTRU   6$                                              ;
                                       0608  C6      4C  AE  9E 00110          MOVAB   FILE_ACL, ACL_BUFFER                    ;
                                    11       66          04  E1 00116 6$:      BBC     #4, INPUT_FLAGS, 7$                     ; 4909
                                                    1C  AE  B5 0011A          TSTW    STATBLK+20                              ;
                                                    0C  13 0011D          BEQL    7$                                      ;
                                                    04  A6  DD 0011F          PUSHL   INPUT_FAB                               ; 4911
                                             00000000G  8F  DD 00122          PUSHL   #BACKUP$_ACCONFLICT                     ;
                                             68 02  FB 00128          CALLS   #2, FILE_ERROR                                  ;
                                    2C   A6      08  AE  10  9C 0012B 7$:      ROTL    #16, STATBLK, INPUT_STATBLK             ; 4916
                                    30   A6      0C  AE  10  9C 00131          ROTL    #16, STATBLK+4, INPUT_STATBLK+4          ; 4917
                                                FDB8  CD  9F 00137          PUSHAB  INPUT_HDR                               ; 4922
                                          FF1C  C7  01  FB 0013B          CALLS   #1, INIT_ATTR                           ;
                                            18  A6  01  D0 00140          MOVL    #1, INPUT_BLOCK                         ; 4928
                                    1C   A6      64  A6  10  9C 00144          ROTL    #16, INPUT_RECATTR+8, INPUT_MAXBLOCK     ; 4929
                                                    68  A6  B5 0014A          TSTW    INPUT_RECATTR+12                        ; 4930
                                                    03  12 0014D          BNEQ    8$                                      ;
                                                    1C  A6  D7 0014F          DECL    INPUT_MAXBLOCK                          ; 4931
                                            30  A6      1C  A6  D1 00152 8$:      CMPL    INPUT_MAXBLOCK, INPUT_STATBLK+4         ; 4933
                                                07  1A 00157          BGTRU   9$                                              ;
                                            F0  8F      5C  A6  93 00159          BITB    INPUT_RECATTR, #240                     ; 4934
                                                05  13 0015E          BEQL    10$                                             ;
                                            1C  A6      30  A6  D0 00160 9$:      MOVL    INPUT_STATBLK+4, INPUT_MAXBLOCK          ; 4936
                                    1A      FF42  C6        03  E1 00165 10$:     BBC     #3, QUAL+10, 11$                        ; 4941
                                                FDB8  CD  9F 0016B          PUSHAB  INPUT_HDR                               ;
                                          F214  C7  01  FB 0016F          CALLS   #1, GET_PLC_DATA                        ;
                                    0B      FF42  C6        03  E1 00174          BBC     #3, QUAL+10, 11$                        ; 4949
                                                57      FF46  C6  E9 0017A          BLBC    QUAL+14, 16$                            ; 4951
                                    22          66          03  E1 0017F          BBC     #3, INPUT_FLAGS, 13$                    ; 4953
                                                            51  11 00183          BRB     16$                                     ;
                                    0A      FF41  C6        06  E1 00185 11$:     BBC     #6, QUAL+9, 12$                         ; 4957
                                                            04  DD 0018B          PUSHL   #4                                      ; 4959
                                                67          01  FB 0018D          CALLS   #1, SELECT_INPUT_FILE                   ;
                                                12          50  E9 00190          BLBC    R0, 13$                                 ;
                                                            41  11 00193          BRB     16$                                     ;
                                                            01  DD 00195 12$:     PUSHL   #1                                      ; 4961
                                                67          01  FB 00197          CALLS   #1, SELECT_INPUT_FILE                   ;
                                                08          50  E9 0019A          BLBC    R0, 13$                                 ;
                                                            04  DD 0019D          PUSHL   #4                                      ; 4965
                                                67          01  FB 0019F          CALLS   #1, SELECT_INPUT_FILE                   ;
                                                31          50  E8 001A2          BLBS    R0, 16$                                 ;
```

```
                                    7E  7C 001A5 13$:     CLRQ    -(SP)                                   ; 4972
                                    7E  7C 001A7           CLRQ    -(SP)                                   ;
                                    7E  7C 001A9           CLRQ    -(SP)                                   ;
                                    7E  7C 001AB           CLRQ    -(SP)                                   ;
                                20  AE  9F 001AD           PUSHAB  IOSB                                    ;
                                34  DD 001B0           PUSHL   #52                                     ;
                                FC  A6  DD 001B2           PUSHL   INPUT_CHAN                              ;
                                    7E  D4 001B5           CLRL    -(SP)                                   ;
                            69      0C  FB 001B7           CALLS   #12, SYS$QIOW                           ;
                            53      50  D0 001BA           MOVL    R0, STATUS                              ;
                            06      53  E9 001BD           BLBC    STATUS, 14$                             ; 4973
                            53      6E  3C 001C0           MOVZWL  IOSB, STATUS                            ;
                            0A      53  E8 001C3           BLBS    STATUS, 15$                             ; 4974
                                    53  DD 001C6 14$:      PUSHL   STATUS                                  ; 4979
                                04  A6  DD 001C8           PUSHL   INPUT_FAB                               ; 4978
                                    5A  DD 001CB           PUSHL   R10                                     ; 4977
                            68      03  FB 001CD           CALLS   #3, FILE_ERROR                          ;
                            66      01  8A 001D0 15$:      BICB2   #1, INPUT_FLAGS                         ; 4980
                                0164    31 001D3           BRW     40$                                     ; 4981
                            06  01D7    C6  E9 001D6 16$:      BLBC    DIR_STATUS, 17$                 ; 4989
                        04  01D7    C6      01  E1 001DB           BBC     #1, DIR_STATUS, 18$             ; 4990
                            B2  A6          04  88 001E1 17$:      BISB2   #4, COM_FLAGS                   ; 4992
                            50      10  A6  D0 001E5 18$:      MOVL    INPUT_QUAL, R0                  ; 4994
                                20  A0  D6 001E9           INCL    32(R0)                                  ;
                            66      20  88 001EC           BISB2   #32, INPUT_FLAGS                        ; 4995
                                FDB8    CD  9F 001EF           PUSHAB  INPUT_HDR                           ; 5000
                        F471    C7      01  FB 001F3           CALLS   #1, FILE_ATTRIBUTES                     ;
                            52          0E  D0 001F8           MOVL    #14, ALWAYS_NOBACK                      ; 5007
                            02  FDBF    CD  91 001FB           CMPB    INPUT_HDR+7, #2                 ; 5008
                                    05  12 00200           BNEQ    19$                                     ;
                            52  0200    8F  A8 00202           BISW2   #512, ALWAYS_NOBACK                     ;
                    0B      58  A6          01  E1 00207 19$:      BBC     #1, INPUT_FILECHAR, 20$         ; 5010
                    05  FF42  C6          01  E0 0020C           BBS     #1, QUAL+T0, 20$                ; 5011
                    1B      59  A6          05  E1 00212           BBC     #5, INPUT_FILECHAR+1, 22$       ; 5012
                            50      08  A6  D0 00217 20$:      MOVL    INPUT_NAM, R0                   ; 5014
                            0F      24  A0  B1 0021B           CMPW    36(R0), #15                             ;
                                    0D  1A 0021F           BGTRU   21$                                     ;
                                29  A0  95 00221           TSTB    41(R0)                                  ; 5015
                                    08  12 00224           BNEQ    21$                                     ;
                            51      24  A0  3C 00226           MOVZWL  36(R0), R1                      ; 5016
                    04              52      51  E0 0022A           BBS     R1, ALWAYS_NOBACK, 22$          ;
                                    54  94 0022E 21$:      CLRB    NOBACK_FLAG                             ; 5018
                                    03  11 00230           BRB     23$                                     ;
                            54          01  90 00232 22$:      MOVB    #1, NOBACK_FLAG                 ; 5020
                    0F  FF42    C6          06  E1 00235 23$:      BBC     #6, QUAL+10, 24$                ; 5025
                    0A      59  A6          05  E0 0023B           BBS     #5, INPUT_FILECHAR+1, 24$       ; 5026
                            7E          54  9A 00240           MOVZBL  NOBACK_FLAG, -(SP)                      ; 5028
                00000000G   00          01  FB 00243           CALLS   #1, WRITE_JOUR_FILE                     ;
                            25          54  E9 0024A 24$:      BLBC    NOBACK_FLAG, 26$                ; 5031
                            50      08  A6  D0 0024D           MOVL    INPUT_NAM, R0                   ; 5039
                            0F      24  A0  B1 00251           CMPW    36(R0), #15                             ;
                                    0D  1A 00255           BGTRU   25$                                     ;
                                29  A0  95 00257           TSTB    41(R0)                                  ; 5040
                                    08  12 0025A           BNEQ    25$                                     ;
                            51      24  A0  3C 0025C           MOVZWL  36(R0), R1                      ; 5041
                    3E              52      51  E0 00260           BBS     R1, ALWAYS_NOBACK, 30$          ;
                                04  A6  DD 00264 25$:      PUSHL   INPUT_FAB                               ; 5042
```

```
                                00000000G   8F   DD 00267          PUSHL   #BACKUP$_NOBACKUP                                ;
                            68              02   FB 0026D          CALLS   #2, FILE_ERROR                                   ;
                                            30   11 00270          BRB     30$                                              ; 5031
                                            54   94 00272 26$:     CLRB    NOBACK_FLAG                                      ; 5048
                            21      FF46    C6   E9 00274          BLBC    QUAL+14, 29$                                     ; 5049
                            52      0088    C6   D0 00279          MOVL    INPUT_VBN_LIST, VBN                              ; 5055
                            50      0088    C6   9E 0027E 27$:     MOVAB   INPUT_VBN_LIST, R0                               ; 5056
                            50              52   D1 00283          CMPL    VBN, R0                                          ;
                                            1A   13 00286          BEQL    30$                                              ;
                      18    A6        0A    A2   7D 00288          MOVQ    10(VBN), INPUT_BLOCK                             ; 5058
                  EEF3      C7              00   FB 0028D 28$:     CALLS   #0, SAVE_BLOCKS                                  ; 5060
                            F8              50   E9 00292          BLBC    R0, 28$                                          ;
                            52              62   D0 00295          MOVL    (VBN), VBN                                       ; 5061
                                            E4   11 00298          BRB     27$                                              ; 5056
                  EEF3      C7              00   FB 0029A 29$:     CALLS   #0, SAVE_BLOCKS                                  ; 5065
                            F8              50   E9 0029F          BLBC    R0, 29$                                          ;
              0C      FF47  C6              06   E1 002A2 30$:     BBC     #6, QUAL+15, 31$                                 ; 5072
                  EC45      C7              00   FB 002A8          CALLS   #0, SAVE_WRITE                                   ; 5075
          00000000G         00              00   FB 002AD          CALLS   #0, CLOSE_RESTORE                                ; 5076
              05  FF42      C6              03   E1 002B4 31$:     BBC     #3, QUAL+10, 32$                                 ; 5082
                  F1DE      C7              00   FB 002BA          CALLS   #0, FREE_PLC_BLOCKS                              ;
                                            7E   7C 002BF 32$:     CLRQ    -(SP)                                            ; 5090
                                            7E   7C 002C1          CLRQ    -(SP)                                            ;
                                            7E   7C 002C3          CLRQ    -(SP)                                            ;
                                            7E   7C 002C5          CLRQ    -(SP)                                            ;
                                20          AE   9F 002C7          PUSHAB  IOSB                                             ;
                                            34   DD 002CA          PUSHL   #52                                              ;
                                FC          A6   DD 002CC          PUSHL   INPUT_CHAN                                       ;
                                            7E   D4 002CF          CLRL    -(SP)                                            ;
                            69              0C   FB 002D1          CALLS   #12, SYS$QIOW                                    ;
                            53              50   D0 002D4          MOVL    R0, STATUS                                       ;
                            06              53   E9 002D7          BLBC    STATUS, 33$                                      ; 5091
                            53              6E   3C 002DA          MOVZWL  IOSB, STATUS                                     ;
                            0A              53   E8 002DD          BLBS    STATUS, 34$                                      ; 5092
                                            53   DD 002E0 33$:     PUSHL   STATUS                                           ; 5097
                                    04      A6   DD 002E2          PUSHL   INPUT_FAB                                        ; 5096
                                            5A   DD 002E5          PUSHL   R10                                              ; 5095
                            68              03   FB 002E7          CALLS   #3, FILE_ERROR                                   ;
                            66              01   8A 002EA 34$:     BICB2   #1, INPUT_FLAGS                                  ; 5098
          20    FF43      C6              01   E1 002ED          BBC     #1, QUAL+11, 37$                                 ; 5103
          1A    FF47      C6              06   E0 002F3          BBS     #6, QUAL+15, 37$                                 ;
                            0B              54   E9 002F9          BLBC    NOBACK_FLAG, 35$                                 ; 5105
                                    04      A6   DD 002FC          PUSHL   INPUT_FAB                                        ; 5109
                                00000000G   8F   DD 002FF          PUSHL   #BACKUP$_HEADCOPIED                              ; 5107
                                            09   11 00305          BRB     36$                                              ;
                                    04      A6   DD 00307 35$:     PUSHL   INPUT_FAB                                        ; 5113
                                00000000G   8F   DD 0030A          PUSHL   #BACKUP$_COPIED                                  ; 5111
                            68              02   FB 00310 36$:     CALLS   #2, FILE_ERROR                                   ;
              0C    FF44      C6              06   E1 00313 37$:     BBC     #6, QUAL+12, 38$                                 ; 5121
                                    01D7    C6   95 00319          TSTB    DIR_STATUS                                       ;
                                            12   13 0031D          BEQL    39$                                              ;
              0C    01D7      C6              01   E0 0031F          BBS     #1, DIR_STATUS, 39$                              ;
              0F    FF46      C6              02   E1 00325 38$:     BBC     #2, QUAL+14, 40$                                 ; 5123
                                    01D7    C6   95 0032B          TSTB    DIR_STATUS                                       ;
                                            09   12 0032F          BNEQ    40$                                              ;
              05                66              05   E1 00331 39$:     BBC     #5, INPUT_FLAGS, 40$                             ; 5125
                        0357    C7              00   FB 00335          CALLS   #0, SAVE_FILE_ID                                 ; 5127
```

```
                                    50          01  D0 0033A 40$:    MOVL    #1, R0                                          ; 5131
                                                04  0033D           RET                                                     ;

; Routine Size:  830 bytes,    Routine Base:  CODE + 19B7
```

```
;  3607     5132  1 %SBTTL 'PHYS_SAVE - main routine for physical save'
;  3608     5133  1 ROUTINE PHYS_SAVE: NOVALUE=
;  3609     5134  1
;  3610     5135  1 !++
;  3611     5136  1 !
;  3612     5137  1 ! FUNCTIONAL DESCRIPTION:
;  3613     5138  1 !       This routine performs a physical save.
;  3614     5139  1 !
;  3615     5140  1 ! INPUT PARAMETERS:
;  3616     5141  1 !       NONE
;  3617     5142  1 !
;  3618     5143  1 ! IMPLICIT INPUTS:
;  3619     5144  1 !       NONE
;  3620     5145  1 !
;  3621     5146  1 ! OUTPUT PARAMETERS:
;  3622     5147  1 !       NONE
;  3623     5148  1 !
;  3624     5149  1 ! IMPLICIT OUTPUTS:
;  3625     5150  1 !       NONE
;  3626     5151  1 !
;  3627     5152  1 ! ROUTINE VALUE:
;  3628     5153  1 !       NONE
;  3629     5154  1 !
;  3630     5155  1 ! SIDE EFFECTS:
;  3631     5156  1 !       NONE
;  3632     5157  1 !
;  3633     5158  1 !--
;  3634     5159  1
;  3635     5160  2 BEGIN
;  3636     5161  2 LOCAL
;  3637     5162  2         INPUT_DEVCHAR:  BBLOCK[DIB$K_LENGTH],   ! Device characteristics
;  3638     5163  2         DESC:           VECTOR[2],              ! Descriptor
;  3639     5164  2         STATUS;                                 ! Status variable
;  3640     5165  2
;  3641     5166  2
;  3642     5167  2 ! Get device characteristics for the input device.
;  3643     5168  2 !
;  3644     5169  2 DESC[0] = DIB$K_LENGTH;
;  3645     5170  2 DESC[1] = INPUT_DEVCHAR;
;  3646     5171  2 STATUS = $GETCHN(CHAN=.INPUT_CHAN, PRIBUF=DESC);
;  3647     5172  2 IF NOT .STATUS
;  3648     5173  2 THEN
;  3649     5174  2     FILE_ERROR(
;  3650     5175  2         BACKUP$_GETCHN,
;  3651     5176  2         .INPUT_FAB,
;  3652     5177  2         .STATUS);
;  3653     5178  2
;  3654     5179  2
;  3655     5180  2 ! Get the bad block data.
;  3656     5181  2 !
;  3657     5182  2 INPUT_BAD = GET_BADBLOCKS(.INPUT_FAB, .INPUT_CHAN, INPUT_DEVCHAR);
;  3658     5183  2
;  3659     5184  2
;  3660     5185  2 ! Write the physical volume attribute record.
;  3661     5186  2 !
;  3662     5187  2 PHYSVOL_SUMMARY(INPUT_DEVCHAR);
;  3663     5188  2
```

```
;3664           5189  2
;3665           5190  2 ! Compute how many blocks to copy.  If the disk has last track bad
;3666           5191  2 ! block data, do not copy the last track.  Then, do the copy.
;3667           5192  2 !
;3668           5193  2 INPUT_BLOCK = 0;
;3669           5194  2 INPUT_MAXBLOCK = .INPUT_DEVCHAR[DIB$L_MAXBLOCK] - 1;
;3670           5195  2 IF NOT .BBLOCK [INPUT_DEVCHAR[DIB$L_DEVCHAR], DEV$V_RCT]
;3671           5196  2 AND .INPUT_DEVCHAR[DIB$L_MAXBLOCK] GTRU SMALL_DISK
;3672           5197  2 THEN
;3673           5198  2     INPUT_MAXBLOCK = .INPUT_MAXBLOCK -
;3674           5199  2         .INPUT_DEVCHAR[DIB$B_SECTORS] /
;3675           5200  4         ((.INPUT_DEVCHAR[DIB$B_SECTORS] *
;3676           5201  4           .INPUT_DEVCHAR[DIB$B_TRACKS] *
;3677           5202  3           .INPUT_DEVCHAR[DIB$W_CYLINDERS]) /
;3678           5203  2           .INPUT_DEVCHAR[DIB$L_MAXBLOCK]);
;3679           5204  2 UNTIL SAVE_BLOCKS() DO 0;
;3680           5205  2
;3681           5206  2
;3682           5207  2 ! If the output is Files-11, release a partially filled buffer to the output
;3683           5208  2 ! procedure.  This serves to synchronize input and output.  Then, if /VERIFY
;3684           5209  2 ! is in effect, do the verification.
;3685           5210  2 !
;3686           5211  2 IF .QUAL[QUAL_OF11]
;3687           5212  2 THEN
;3688           5213  3     BEGIN
;3689           5214  3     SAVE_WRITE();
;3690           5215  3     CLOSE_RESTORE();
;3691           5216  2     END;
;3692           5217  2
;3693           5218  2
;3694           5219  2 ! Log the copy if requested.
;3695           5220  2 !
;3696           5221  2 IF .QUAL[QUAL_LOG] AND NOT .QUAL[QUAL_OF11]
;3697           5222  2 THEN
;3698           5223  2     FILE_ERROR(
;3699           5224  2         BACKUP$_COPIED,
;3700           5225  2         .INPUT_FAB);
;3701           5226  1 END;


                                                        .EXTRN  SYS$GETCHN

                                        003C 00000 PHYS_SAVE:
                                                        .WORD   Save R2,R3,R4,R5                        ;5133
                        55 00000000G    00    9E 00002  MOVAB   FILE_ERROR, R5                        ;
                        54 00000000'    EF    9E 00009  MOVAB   INPUT_FAB, R4                         ;
                        5E        88    AE    9E 00010  MOVAB   -120(SP), SP                          ;
                        7E        74    8F    9A 00014  MOVZBL  #116, DESC                            ;5169
                  04    AE        08    AE    9E 00018  MOVAB   INPUT_DEVCHAR, DESC+4                 ;5170
                                        7E    7C 0001D  CLRQ    -(SP)                                 ;5171
                                  08    AE    9F 0001F  PUSHAB  DESC                                  ;
                                        7E    D4 00022  CLRL    -(SP)                                 ;
                                  F8    A4    DD 00024  PUSHL   INPUT_CHAN                            ;
                  00000000G     00          05    FB 00027  CALLS   #5, SYS$GETCHN                    ;
                                0D          50    E8 0002E  BLBS    STATUS, 1$                        ;5172
                                            50    DD 00031  PUSHL   STATUS                            ;5177
```

```
                                                64  DD 00033          PUSHL   INPUT_FAB                                ; 5176
                                    00000000G   8F  DD 00035          PUSHL   #BACKUP$_GETCHN                          ; 5174
                                            65      03  FB 0003B          CALLS   #3, FILE_ERROR                           ;
                                                08  AE  9F 0003E 1$:      PUSHAB  INPUT_DEVCHAR                            ; 5182
                                                F8  A4  DD 00041          PUSHL   INPUT_CHAN                               ;
                                                64  DD 00044          PUSHL   INPUT_FAB                                ;
                                00000000G       00      03  FB 00046          CALLS   #3, GET_BADBLOCKS                        ;
                                        10      A4      50  D0 0004D          MOVL    R0, INPUT_BAD                            ;
                                                08  AE  9F 00051          PUSHAB  INPUT_DEVCHAR                            ; 5187
                                    F57A        CF      01  FB 00054          CALLS   #1, PHYSVOL_SUMMARY                      ;
                                                14  A4  D4 00059          CLRL    INPUT_BLOCK                              ; 5193
                                18  A4      78  AE      01  C3 0005C          SUBL3   #1, INPUT_DEVCHAR+112, INPUT_MAXBLOCK    ; 5194
                                            2C  09  AE  E8 00062          BLBS    INPUT_DEVCHAR+1, 2$                      ; 5195
                                00001000    8F      78  AE  D1 00066          CMPL    INPUT_DEVCHAR+112, #4096                 ; 5196
                                                22  1B 0006E          BLEQU   2$                                       ;
                                            50  10  AE  9A 00070          MOVZBL  INPUT_DEVCHAR+8, R0                      ; 5201
                                            51  11  AE  9A 00074          MOVZBL  INPUT_DEVCHAR+9, R1                      ;
                                            50      51  C4 00078          MULL2   R1, R0                                   ;
                                            52  12  AE  3C 0007B          MOVZWL  INPUT_DEVCHAR+10, R2                     ; 5202
                                            50      52  C4 0007F          MULL2   R2, R0                                   ;
                                            50  78  AE  C6 00082          DIVL2   INPUT_DEVCHAR+112, R0                    ; 5203
                                            53  10  AE  9A 00086          MOVZBL  INPUT_DEVCHAR+8, R3                      ; 5200
                            50              53      50  C7 0008A          DIVL3   R0, R3, R0                               ;
                                    18      A4      50  C2 0008E          SUBL2   R0, INPUT_MAXBLOCK                       ; 5199
                                E6FB        CF      00  FB 00092 2$:      CALLS   #0, SAVE_BLOCKS                          ; 5204
                                            F8      50  E9 00097          BLBC    R0, 2$                                   ;
                            0C      FF43    C4      06  E1 0009A          BBC     #6, QUAL+15, 3$                          ; 5211
                                E43F        CF      00  FB 000A0          CALLS   #0, SAVE_WRITE                           ; 5214
                                00000000G   00      00  FB 000A5          CALLS   #0, CLOSE_RESTORE                        ; 5215
                            11      FF3F    C4      01  E1 000AC 3$:      BBC     #1, QUAL+11, 4$                          ; 5221
                            0B      FF43    C4      06  E0 000B2          BBS     #6, QUAL+15, 4$                          ;
                                                64  DD 000B8          PUSHL   INPUT_FAB                                ; 5225
                                    00000000G   8F  DD 000BA          PUSHL   #BACKUP$_COPIED                          ; 5223
                                            65      02  FB 000C0          CALLS   #2, FILE_ERROR                           ;
                                                    04 000C3 4$:      RET                                              ; 5226

; Routine Size:  196 bytes,    Routine Base:  CODE + 1CF5
```

```
;  3703                 5227  1 %SBTTL 'RESTARTABLE_CONDITION - evaluate error for restartability'
;  3704                 5228  1 GLOBAL ROUTINE RESTARTABLE_CONDITION(COND)=
;  3705                 5229  1
;  3706                 5230  1 !++
;  3707                 5231  1 !
;  3708                 5232  1 ! FUNCTIONAL DESCRIPTION:
;  3709                 5233  1 !       This routine evaluates whether the status value is eligible for a
;  3710                 5234  1 !       restart and whether the last checkpoint was a valid one.
;  3711                 5235  1 !
;  3712                 5236  1 ! INPUT PARAMETERS:
;  3713                 5237  1 !       COND            - Condition value
;  3714                 5238  1 !
;  3715                 5239  1 ! IMPLICIT INPUTS:
;  3716                 5240  1 !       NONE
;  3717                 5241  1 !
;  3718                 5242  1 ! OUTPUT PARAMETERS:
;  3719                 5243  1 !       NONE
;  3720                 5244  1 !
;  3721                 5245  1 ! IMPLICIT OUTPUTS:
;  3722                 5246  1 !       NONE
;  3723                 5247  1 !
;  3724                 5248  1 ! ROUTINE VALUE:
;  3725                 5249  1 !       True if the condition is eligible for restart.
;  3726                 5250  1 !
;  3727                 5251  1 ! SIDE EFFECTS:
;  3728                 5252  1 !       NONE
;  3729                 5253  1 !
;  3730                 5254  1 !--
;  3731                 5255  1
;  3732                 5256  2 BEGIN
;  3733                 5257  2 MAP
;  3734                 5258  2         COND:           BLOCK;          ! Condition value
;  3735                 5259  2 EXTERNAL LITERAL
;  3736                 5260  2         BACKUP$_WAITIDLEBCB,
;  3737                 5261  2         BACKUP$_BUFFERSLOST,
;  3738                 5262  2         BACKUP$_FREEBADBUFF,
;  3739                 5263  2         BACKUP$_ACTIVEBCB,
;  3740                 5264  2         BACKUP$_ALLOCMEM,
;  3741                 5265  2         BACKUP$_FREEMEM,
;  3742                 5266  2         BACKUP$_OPERFAIL,
;  3743                 5267  2         BACKUP$_MAXVOLS,
;  3744                 5268  2         BACKUP$_PROCINDEX;
;  3745                 5269  2 BIND
;  3746                 5270  2         TABLE = PLIT(
;  3747                 5271  2             BACKUP$_WAITIDLEBCB,
;  3748                 5272  2             BACKUP$_BUFFERSLOST,
;  3749                 5273  2             BACKUP$_FREEBADBUFF,
;  3750                 5274  2             BACKUP$_ACTIVEBCB,
;  3751                 5275  2             BACKUP$_ALLOCMEM,
;  3752                 5276  2             BACKUP$_FREEMEM,
;  3753                 5277  2             BACKUP$_OPERFAIL,
;  3754                 5278  2             BACKUP$_MAXVOLS,
;  3755                 5279  2             BACKUP$_PROCINDEX)
;  3756                 5280  2         : VECTOR;
;  3757                 5281  2
;  3758                 5282  2
;  3759                 5283  2 IF
```

```
;  3760          5284  2      .COND[STS$V_SEVERITY] EQL STS$K_SEVERE AND
;  3761          5285  2      .COND[STS$V_FAC_NO] EQL BACKUP$_FACILITY AND
;  3762          5286  2      NOT .COM_FLAGS[COM_DSBL_RSTRT] AND
;  3763          5287  3      (DECR X FROM .TABLE[-1]=1 TO 0 DO
;  3764          5288  3          IF .COND EQL .TABLE[.X] THEN EXITLOOP FALSE)
;  3765          5289  2  THEN
;  3766          5290  2      TRUE
;  3767          5291  2  ELSE
;  3768          5292  2      FALSE
;  3769          5293  1  END;


                                                        01DB9          .BLKB   3
                                                00000009  01DBC        .LONG   9                                              ;
00000000G 00000000G 00000000G 00000000G 00000000G 00000000G 01DC0 P.AAJ:  .LONG   BACKUP$_WAITIDLEBCB, BACKUP$_BUFFERSLOST, -    ;
                          00000000G 00000000G 00000000G 01DD8                  BACKUP$_FREEBADBUFF, BACKUP$_ACTIVEBCB, -     ;
                                                                               BACKUP$_ALLOCMEM, BACKUP$_FREEMEM, -          ;
                                                                               BACKUP$_OPERFAIL, BACKUP$_MAXVOLS, -          ;
                                                                               BACKUP$_PROCINDEX                             ;

                                                                TABLE=              P.AAJ
                                                                      .EXTRN   BACKUP$_WAITIDLEBCB
                                                                      .EXTRN   BACKUP$_BUFFERSLOST
                                                                      .EXTRN   BACKUP$_FREEBADBUFF
                                                                      .EXTRN   BACKUP$_ALLOCMEM
                                                                      .EXTRN   BACKUP$_FREEMEM
                                                                      .EXTRN   BACKUP$_OPERFAIL
                                                                      .EXTRN   BACKUP$_MAXVOLS
                                                                      .EXTRN   BACKUP$_PROCINDEX

                                                          0000 00000          .ENTRY  RESTARTABLE_CONDITION, Save nothing     ; 5228
            04        04   AC            03              00  ED 00002          CMPZV   #0, #3, COND, #4                        ; 5284
                                                        29  12 00008          BNEQ    3$                                      ;
00000000G   8F        06   AC            0C              00  ED 0000A          CMPZV   #0, #12, COND+2, #BACKUP$_FACILITY      ; 5285
                                                        1D  12 00014          BNEQ    3$                                      ;
                                                00000000' EF  95 00016          TSTB    COM_FLAGS                               ; 5286
                                                        15  19 0001C          BLSS    3$                                      ;
                                              50        B7  AF  D0 0001E          MOVL    TABLE-4, X                              ; 5288
                                                        08  11 00022          BRB     2$                                      ;
                                           B2 AF40        04  AC  D1 00024 1$:      CMPL    COND, TABLE[X]                          ;
                                                        07  13 0002A          BEQL    3$                                      ;
                                              F5        50  F4 0002C 2$:      SOBGEQ  X, 1$                                   ;
                                              50        01  D0 0002F          MOVL    #1, R0                                  ; 5283
                                                            04 00032          RET                                             ;
                                                        50  D4 00033 3$:      CLRL    R0                                      ;
                                                            04 00035          RET                                             ; 5293

; Routine Size:  54 bytes,    Routine Base:  CODE + 1DE4
```

```
;3771           5294  1 %SBTTL 'SAVE_HANDLER - condition handler for SAVE'
;3772           5295  1 ROUTINE SAVE_HANDLER(SIG,MECH)=
;3773           5296  1
;3774           5297  1 !++
;3775           5298  1 !
;3776           5299  1 ! FUNCTIONAL DESCRIPTION:
;3777           5300  1 !       This routine is established as a handler for routine SAVE.  It
;3778           5301  1 !       determines the recovery from a fatal signal.
;3779           5302  1 !
;3780           5303  1 ! INPUT PARAMETERS:
;3781           5304  1 !       SIG             - Standard VMS condition handler parameters.
;3782           5305  1 !       MECH            -
;3783           5306  1 !
;3784           5307  1 ! IMPLICIT INPUTS:
;3785           5308  1 !       NONE
;3786           5309  1 !
;3787           5310  1 ! OUTPUT PARAMETERS:
;3788           5311  1 !       NONE
;3789           5312  1 !
;3790           5313  1 ! IMPLICIT OUTPUTS:
;3791           5314  1 !       NONE
;3792           5315  1 !
;3793           5316  1 ! ROUTINE VALUE:
;3794           5317  1 !       VMS status code (SS$_CONTINUE or SS$_RESIGNAL).
;3795           5318  1 !       However, if SAVE_RESTART is called by GET_RESTART,
;3796           5319  1 !       control never returns.
;3797           5320  1 !
;3798           5321  1 ! SIDE EFFECTS:
;3799           5322  1 !       NONE
;3800           5323  1 !
;3801           5324  1 !--
;3802           5325  1
;3803           5326  2 BEGIN
;3804           5327  2 MAP
;3805           5328  2         SIG:            REF BBLOCK,     ! Signal parameters
;3806           5329  2         MECH:           REF BBLOCK;     ! Mechanism parameters
;3807           5330  2
;3808           5331  2
;3809           5332  2 IF RESTARTABLE_CONDITION(.SIG[CHF$L_SIG_NAME])
;3810           5333  2 THEN
;3811           5334  3     BEGIN
;3812           5335  3     IF
;3813           5336  3         .SIG[CHF$L_SIG_NAME] EQL BACKUP$_WRITERRS OR
;3814           5337  3         .SIG[CHF$L_SIG_NAME] EQL BACKUP$_FATALERR
;3815           5338  3     THEN
;3816           5339  3         IF .RWSV_VOL_NUMBER EQL 1
;3817           5340  3         THEN
;3818           5341  3             GET_RESTART(.SIG, %B'010')
;3819           5342  3         ELSE
;3820           5343  3             GET_RESTART(.SIG, %B'011')
;3821           5344  3     ELSE
;3822           5345  3         IF .RWSV_VOL_NUMBER EQL 1
;3823           5346  3         THEN
;3824           5347  3             RETURN SS$_RESIGNAL
;3825           5348  3         ELSE
;3826           5349  3             GET_RESTART(.SIG, %B'001');
;3827           5350  3
```

```
;  3828          5351  3
;  3829          5352  3      RETURN SS$_CONTINUE;
;  3830          5353  2      END;
;  3831          5354  2
;  3832          5355  2
;  3833          5356  2 SS$_RESIGNAL
;  3834          5357  1 END;


                                        000C 00000 SAVE_HANDLER:
                                                   .WORD   Save R2,R3                                      ; 5295
                          53 00000000'  EF  9E 00002       MOVAB   RWSV_VOL_NUMBER, R3
                          52        04  AC  D0 00009       MOVL    SIG, R2                                 ; 5332
                                    04  A2  DD 0000D       PUSHL   4(R2)
                                B6  AF        01  FB 00010       CALLS   #1, RESTARTABLE_CONDITION
                                    35        50  E9 00014       BLBC    R0, 5$
                     00000000G  8F        04  A2  D1 00017       CMPL    4(R2), #BACKUP$_WRITERRS        ; 5336
                                              0A  13 0001F       BEQL    1$
                     00000000G  8F        04  A2  D1 00021       CMPL    4(R2), #BACKUP$_FATALERR        ; 5337
                                              0D  12 00029       BNEQ    3$
                                01            63  B1 0002B 1$:   CMPW    RWSV_VOL_NUMBER, #1             ; 5339
                                              04  12 0002E       BNEQ    2$
                                              02  DD 00030       PUSHL   #2                              ; 5341
                                              0B  11 00032       BRB     4$
                                              03  DD 00034 2$:   PUSHL   #3                              ; 5343
                                              07  11 00036       BRB     4$
                                01            63  B1 00038 3$:   CMPW    RWSV_VOL_NUMBER, #1             ; 5345
                                              0F  13 0003B       BEQL    5$
                                              01  DD 0003D       PUSHL   #1                              ; 5349
                                              52  DD 0003F 4$:   PUSHL   R2
                     00000000G  00            02  FB 00041       CALLS   #2, GET_RESTART
                                50            01  D0 00048       MOVL    #1, R0                          ; 5352
                                                  04 0004B       RET
                                50      0918  8F  3C 0004C 5$:   MOVZWL  #2328, R0                       ; 5357
                                                  04 00051       RET

; Routine Size:  82 bytes,    Routine Base:  CODE + 1E1A
```

```
;  3836          5358  1 %SBTTL 'SAVE - main routine for save operation'
;  3837          5359  1 GLOBAL ROUTINE SAVE: NOVALUE=
;  3838          5360  1
;  3839          5361  1 !++
;  3840          5362  1 !
;  3841          5363  1 ! FUNCTIONAL DESCRIPTION:
;  3842          5364  1 !       This routine saves Files-11 media on backup media.
;  3843          5365  1 !
;  3844          5366  1 ! INPUT PARAMETERS:
;  3845          5367  1 !       NONE
;  3846          5368  1 !
;  3847          5369  1 ! IMPLICIT INPUTS:
;  3848          5370  1 !       NONE
;  3849          5371  1 !
;  3850          5372  1 ! OUTPUT PARAMETERS:
;  3851          5373  1 !       NONE
;  3852          5374  1 !
;  3853          5375  1 ! IMPLICIT OUTPUTS:
;  3854          5376  1 !       NONE
;  3855          5377  1 !
;  3856          5378  1 ! ROUTINE VALUE:
;  3857          5379  1 !       NONE
;  3858          5380  1 !
;  3859          5381  1 ! SIDE EFFECTS:
;  3860          5382  1 !       NONE
;  3861          5383  1 !
;  3862          5384  1 !--
;  3863          5385  1
;  3864          5386  2 BEGIN
;  3865          5387  2 BUILTIN
;  3866          5388  2         FP;
;  3867          5389  2
;  3868          5390  2
;  3869          5391  2 CHKPT_HIGH_SP = .FP;                    ! Save upper stack region address
;  3870          5392  2 IF .QUAL[QUAL_OSAV]
;  3871          5393  2 THEN
;  3872          5394  3     BEGIN
;  3873          5395  3     !
;  3874          5396  3     ! Initialize for a save.
;  3875          5397  3     !
;  3876          5398  3     .FP = SAVE_HANDLER;                ! Establish handler
;  3877          5399  3     INIT_BUFFERS(
;  3878          5400  3         .QUAL[QUAL_BUFF_VALUE] + (.QUAL[QUAL_GROU_VALUE] NEQ 0),
;  3879          5401  3         .QUAL[QUAL_BLOC_VALUE]);
;  3880          5402  3     INIT_OUT_SAVE(FALSE);
;  3881          5403  3     IF .QUAL[QUAL_JOUR]
;  3882          5404  3     THEN
;  3883          5405  4         BEGIN
;  3884          5406  4         WRITE_JOUR_SSNAME();
;  3885          5407  4         WRITE_JOUR_VOLUME();
;  3886          5408  3         END;
;  3887          5409  3     IF .QUAL[QUAL_SS_ENCRYP] AND CRYPTO_INIENC NEQA 0
;  3888          5410  3     THEN
;  3889          5411  3         CRYPTO_INIENC() ;
;  3890          5412  3     END
;  3891          5413  2 ELSE
;  3892          5414  3     BEGIN
```

```
;3893   5415  3      !
;3894   5416  3      ! Initialize for a copy or a Files-11 to Files-11 compare.
;3895   5417  3      !
;3896   5418  3      INIT_BUFFERS(
;3897   5419  4          (IF .QUAL[QUAL_COMP] OR .QUAL[QUAL_VERI]
;3898   5420  4              THEN COPY_BUFF_COUNT*2
;3899   5421  3              ELSE COPY_BUFF_COUNT),
;3900   5422  3          COPY_BUFF_SIZE);
;3901   5423  3      INIT_RESTORE();
;3902   5424  3      RWSV_SAVE_QUAL = .QUAL[QUAL_OUTP_LIST];
;3903   5425  2      END;
;3904   5426  2
;3905   5427  2
;3906   5428  2  ! Generate the backup summary record.  If this is an image save, it is done by
;3907   5429  2  ! VOLUME_ATTRIBUTES on the first call, because we need some information
;3908   5430  2  ! from the home block.
;3909   5431  2  !
;3910   5432  2  IF NOT .QUAL[QUAL_IMAG] THEN BACKUP_SUMMARY();
;3911   5433  2
;3912   5434  2
;3913   5435  2  ! Loop over all input specifications.
;3914   5436  2  !
;3915   5437  2  INPUT_QUAL = .QUAL[QUAL_INPU_LIST];
;3916   5438  2  INPUT_NAM = 0;
;3917   5439  2  WHILE .INPUT_QUAL NEQ 0 DO
;3918   5440  3      BEGIN
;3919   5441  3      LOCAL
;3920   5442  3          STATUS;
;3921   5443  3
;3922   5444  3
;3923   5445  3      ! Establish pointers to the interesting control blocks.
;3924   5446  3      ! Strip everything but the device from the expanded string.
;3925   5447  3      !
;3926   5448  3      INPUT_FAB = .INPUT_QUAL[QUAL_PARA_FC];
;3927   5449  3      INPUT_NAM = INPUT_FAB[FC_NAM];
;3928   5450  3      INPUT_NAM[NAM$B_ESL] = .BBLOCK[INPUT_QUAL[QUAL_DEV_DESC], DSC$W_LENGTH];
;3929   5451  3
;3930   5452  3
;3931   5453  3      ! Assign a channel to the input device.
;3932   5454  3      !
;3933   5455  3      STATUS = ASSIGN_INPUT_CHANNEL(INPUT_QUAL[QUAL_DEV_DESC], INPUT_CHAN, 0, 0);
;3934   5456  3      IF NOT .STATUS
;3935   5457  3      THEN
;3936   5458  3          FILE_ERROR(
;3937   5459  3              BACKUP$_OPENIN + STS$K_SEVERE,
;3938   5460  3              .INPUT_FAB,
;3939   5461  3              .STATUS);
;3940   5462  3
;3941   5463  3
;3942   5464  3      IF .QUAL[QUAL_PHYS]
;3943   5465  3      THEN
;3944   5466  3          PHYS_SAVE()
;3945   5467  3      ELSE
;3946   5468  4          BEGIN
;3947   5469  4          COM_FLAGS[COM_FILESEEN] = FALSE;
;3948   5470  4          IF .QUAL[QUAL_FAST]
;3949   5471  4              THEN FAST_FILE_SCAN()
```

```
;  3950          5472  4              ELSE SLOW_FILE_SCAN();
;  3951          5473  4          IF NOT .COM_FLAGS[COM_FILESEEN]
;  3952          5474  4          THEN
;  3953          5475  4              SIGNAL(BACKUP$_NOFILES, 1, INPUT_QUAL[QUAL_EXP_DESC]);
;  3954          5476  3          END;
;  3955          5477  3
;  3956          5478  3
;  3957          5479  3      ! Deassign the input channel.
;  3958          5480  3      !
;  3959          5481  3      $DASSGN(CHAN=.INPUT_CHAN);
;  3960          5482  3      INPUT_CHAN = 0;
;  3961          5483  3
;  3962          5484  3
;  3963          5485  3      IF .COM_FLAGS[COM_STANDALONE] THEN EXITLOOP;
;  3964          5486  3      INPUT_QUAL = .INPUT_QUAL[QUAL_NEXT];
;  3965          5487  2      END;
;  3966          5488  2
;  3967          5489  2
;  3968          5490  2  ! Finish up.
;  3969          5491  2  !
;  3970          5492  2  SAVE_WRITE();
;  3971          5493  2  IF .QUAL[QUAL_OSAV]
;  3972          5494  2  THEN
;  3973          5495  3      BEGIN
;  3974          5496  3      IF .QUAL[QUAL_JOUR]
;  3975          5497  3      THEN
;  3976          5498  4          BEGIN
;  3977          5499  4          WRITE_JOUR_DIRECTORY();
;  3978          5500  4          WRITE_JOUR_VOLUME();
;  3979          5501  4          WRITE_JOUR_SSNAME();
;  3980          5502  3          END;
;  3981          5503  3      FIN_OUT_SAVE(FALSE);
;  3982          5504  3      END
;  3983          5505  2  ELSE
;  3984          5506  2      FIN_RESTORE();
;  3985          5507  2
;  3986          5508  2
;  3987          5509  2  ! Postprocess successfully processed files if requested.
;  3988          5510  2  !
;  3989          5511  2  IF .QUAL[QUAL_RECO] OR .QUAL[QUAL_DELE]
;  3990          5512  2  THEN
;  3991          5513  2      POSTPROCESS_FILES();
;  3992          5514  1  END;


                                    007C 00000          .ENTRY  SAVE, Save R2,R3,R4,R5,R6                    ; 5359
                          56 00000000G  00   9E 00002   MOVAB   CRYPTO_INIENC, R6                             ;
                          55 00000000G  00   9E 00009   MOVAB   WRITE_JOUR_VOLUME, R5                         ;
                          54 00000000G  00   9E 00010   MOVAB   WRITE_JOUR_SSNAME, R4                         ;
                          53 00000000G  00   9E 00017   MOVAB   INIT_BUFFERS, R3                             ;
                          52 00000000'  EF   9E 0001E   MOVAB   QUAL+12, R2                                  ;
                    0260  C2            5D   D0 00025   MOVL    FP, CHKPT_HIGH_SP                            ; 5391
                                  03    A2   95 0002A   TSTB    QUAL+15                                      ; 5392
                                        3F   18 0002D   BGEQ    3$                                           ;
```

```
                        6D      FF7B    CF   9E 0002F          MOVAB   SAVE_HANDLER, (FP)                          ; 5398
                        7E        3C    A2   3C 00034          MOVZWL  QUAL+72, -(SP)                            ; 5401
                        50        40    A2   9A 00038          MOVZBL  QUAL+76, R0                               ; 5400
                                        51   D4 0003C          CLRL    R1                                        ;
                                      42    A2   95 0003E          TSTB    QUAL+78                                   ;
                                        02   13 00041          BEQL    1$                                        ;
                                        51   D6 00043          INCL    R1                                        ;
                                    6140    9F   00045 1$:      PUSHAB  (R1)[R0]                                  ;
                        63              02   FB 00048          CALLS   #2, INIT_BUFFERS                          ;
                                        7E   D4 0004B          CLRL    -(SP)                                     ; 5402
                00000000G       00      01   FB 0004D          CALLS   #1, INIT_OUT_SAVE                         ;
            06        FE    A2          06   E1 00054          BBC     #6, QUAL+10, 2$                           ; 5403
                        64              00   FB 00059          CALLS   #0, WRITE_JOUR_SSNAME                     ; 5406
                        65              00   FB 0005C          CALLS   #0, WRITE_JOUR_VOLUME                     ; 5407
            2E        02    A2          04   E1 0005F 2$:      BBC     #4, QUAL+14, 7$                           ; 5409
                        50              66   9E 00064          MOVAB   CRYPTO_INIENC, R0                         ;
                                        29   13 00067          BEQL    7$                                        ;
                        66              00   FB 00069          CALLS   #0, CRYPTO_INIENC                         ; 5411
                                        24   11 0006C          BRB     7$                                        ; 5392
                        7E      8110    8F   3C 0006E 3$:      MOVZWL  #33040, -(SP)                             ; 5418
                                  FC    A2   95 00073          TSTB    QUAL+8                                    ; 5419
                                        05   19 00076          BLSS    4$                                        ;
                                      01    A2   95 00078          TSTB    QUAL+13                                   ;
                                        04   18 0007B          BGEQ    5$                                        ;
                                        04   DD 0007D 4$:      PUSHL   #4                                        ; 5420
                                        02   11 0007F          BRB     6$                                        ;
                                        02   DD 00081 5$:      PUSHL   #2                                        ; 5419
                        63              02   FB 00083 6$:      CALLS   #2, INIT_BUFFERS                          ;
                00000000G       00      00   FB 00086          CALLS   #0, INIT_RESTORE                          ; 5423
                      9C    A2        F8   A2   D0 0008D          MOVL    QUAL+4, RWSV_SAVE_QUAL                    ; 5424
            05        FE    A2          03   E0 00092 7$:      BBS     #3, QUAL+10, 8$                           ; 5432
                    E5C0    CF          00   FB 00097          CALLS   #0, BACKUP_SUMMARY                        ;
                    00CC    C2      F4    A2   D0 0009C 8$:      MOVL    QUAL, INPUT_QUAL                          ; 5437
                                  00C4    C2   D4 000A2          CLRL    INPUT_NAM                                 ; 5438
                        51        00CC    C2   D0 000A6 9$:      MOVL    INPUT_QUAL, R1                            ; 5439
                                        03   12 000AB          BNEQ    10$                                       ;
                                    009E    31   000AD          BRW     16$                                       ;
                    00C0    C2        04    A1   D0 000B0 10$:     MOVL    4(R1), INPUT_FAB                          ; 5448
    00C4    C2      00C0    C2  00000094    8F   C1 000B6          ADDL3   #148, INPUT_FAB, INPUT_NAM                ; 5449
                        50      00C4    C2   D0 000C2          MOVL    INPUT_NAM, R0                             ; 5450
                      0B    A0        10    A1   90 000C7          MOVB    16(R1), 11(R0)                            ;
                                        7E   7C 000CC          CLRQ    -(SP)                                     ; 5455
                                  00B8    C2   9F 000CE          PUSHAB  INPUT_CHAN                                ;
                                      10    A1   9F 000D2          PUSHAB  16(R1)                                    ;
                00000000G       00      04   FB 000D5          CALLS   #4, ASSIGN_INPUT_CHANNEL                  ;
                        13              50   E8 000DC          BLBS    STATUS, 11$                               ; 5456
                                        50   DD 000DF          PUSHL   STATUS                                    ; 5461
                                  00C0    C2   DD 000E1          PUSHL   INPUT_FAB                                 ; 5460
                              00000000G   8F   DD 000E5          PUSHL   #BACKUP$_OPENIN+4                         ; 5459
                00000000G       00      03   FB 000EB          CALLS   #3, FILE_ERROR                            ;
            07                  62      05   E1 000F2 11$:      BBC     #5, QUAL+12, 12$                          ; 5464
                    FD8E    CF          00   FB 000F6          CALLS   #0, PHYS_SAVE                             ; 5466
                                        33   11 000FB          BRB     15$                                       ;
                      6E    A2          04   8A 000FD 12$:      BICB2   #4, COM_FLAGS                             ; 5469
            09        FD    A2          06   E1 00101          BBC     #6, QUAL+9, 13$                           ; 5470
                00000000G       00      00   FB 00106          CALLS   #0, FAST_FILE_SCAN                        ; 5471
                                        07   11 0010D          BRB     14$                                       ;
```

```
                    00000000G  00             00  FB 0010F 13$:    CALLS   #0, SLOW_FILE_SCAN                          ; 5472
                15        6E   A2             02  E0 00116 14$:    BBS     #2, COM_FLAGS, 15$                          ; 5473
                7E      00CC   C2             08  C1 00118          ADDL3   #8, INPUT_QUAL, -(SP)                       ; 5475
                                               01  DD 00121          PUSHL   #1                                          ;
                                 00000000G     8F  DD 00123          PUSHL   #BACKUP$_NOFILES                            ;
                    00000000G  00             03  FB 00129          CALLS   #3, LIB$SIGNAL                              ;
                                      00B8     C2  DD 00130 15$:     PUSHL   INPUT_CHAN                                  ; 5481
                    00000000G  00             01  FB 00134          CALLS   #1, SYS$DASSGN                              ;
                                      00B8     C2  D4 00138          CLRL    INPUT_CHAN                                  ; 5482
                0A        6E   A2             01  E0 0013F          BBS     #1, COM_FLAGS, 16$                          ; 5485
                        00CC   C2       00CC  D2  D0 00144          MOVL    @INPUT_QUAL, INPUT_QUAL                     ; 5486
                                           FF58 31  00148          BRW     9$                                          ; 5439
                        E21A   CF             00  FB 0014E 16$:     CALLS   #0, SAVE_WRITE                              ; 5492
                                         03   A2  95 00153          TSTB    QUAL+15                                     ; 5493
                                               1D  18 00156          BGEQ    18$                                         ;
                0D        FE   A2             06  E1 00158          BBC     #6, QUAL+10, 17$                            ; 5496
                    00000000G  00             00  FB 0015D          CALLS   #0, WRITE_JOUR_DIRECTORY                    ; 5499
                               65             00  FB 00164          CALLS   #0, WRITE_JOUR_VOLUME                       ; 5500
                               64             00  FB 00167          CALLS   #0, WRITE_JOUR_SSNAME                       ; 5501
                                               7E  D4 0016A 17$:     CLRL    -(SP)                                       ; 5503
                    00000000G  00             01  FB 0016C          CALLS   #1, FIN_OUT_SAVE                            ;
                                               07  11 00173          BRB     19$                                         ; 5493
                    00000000G  00             00  FB 00175 18$:     CALLS   #0, FIN_RESTORE                             ; 5506
                05              62             06  E0 0017C 19$:     BBS     #6, QUAL+12, 20$                            ; 5511
                05        02   A2             02  E1 00180          BBC     #2, QUAL+14, 21$                            ;
                        F70E   CF             00  FB 00185 20$:     CALLS   #0, POSTPROCESS_FILES                       ; 5513
                                               04  0018A 21$:     RET                                                 ; 5514

; Routine Size:  395 bytes,    Routine Base:  CODE + 1E6C
```

```
; 3994          5515  1 END
; 3995          5516  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL

;                                  PSECT SUMMARY

;       Name                    Bytes                           Attributes

;  COMMON                        2124  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  OVR,NOPIC,ALIGN(2)
;  CODE                          8183  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        214         1        1000        00:01.9


;                                    COMMAND QUALIFIERS

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

; Size:          7999 code + 2308 data bytes
; Run Time:         03:05.4
; Elapsed Time:     09:56.3
; Lines/CPU Min:     1785
; Lexemes/CPU-Min: 44378
; Memory Used:  743 pages
; Compilation Complete
```

SAVE
LIS

RESTORE
LIS

RESTARTM
LIS

STAACP
LIS

STAACCTL
LIS