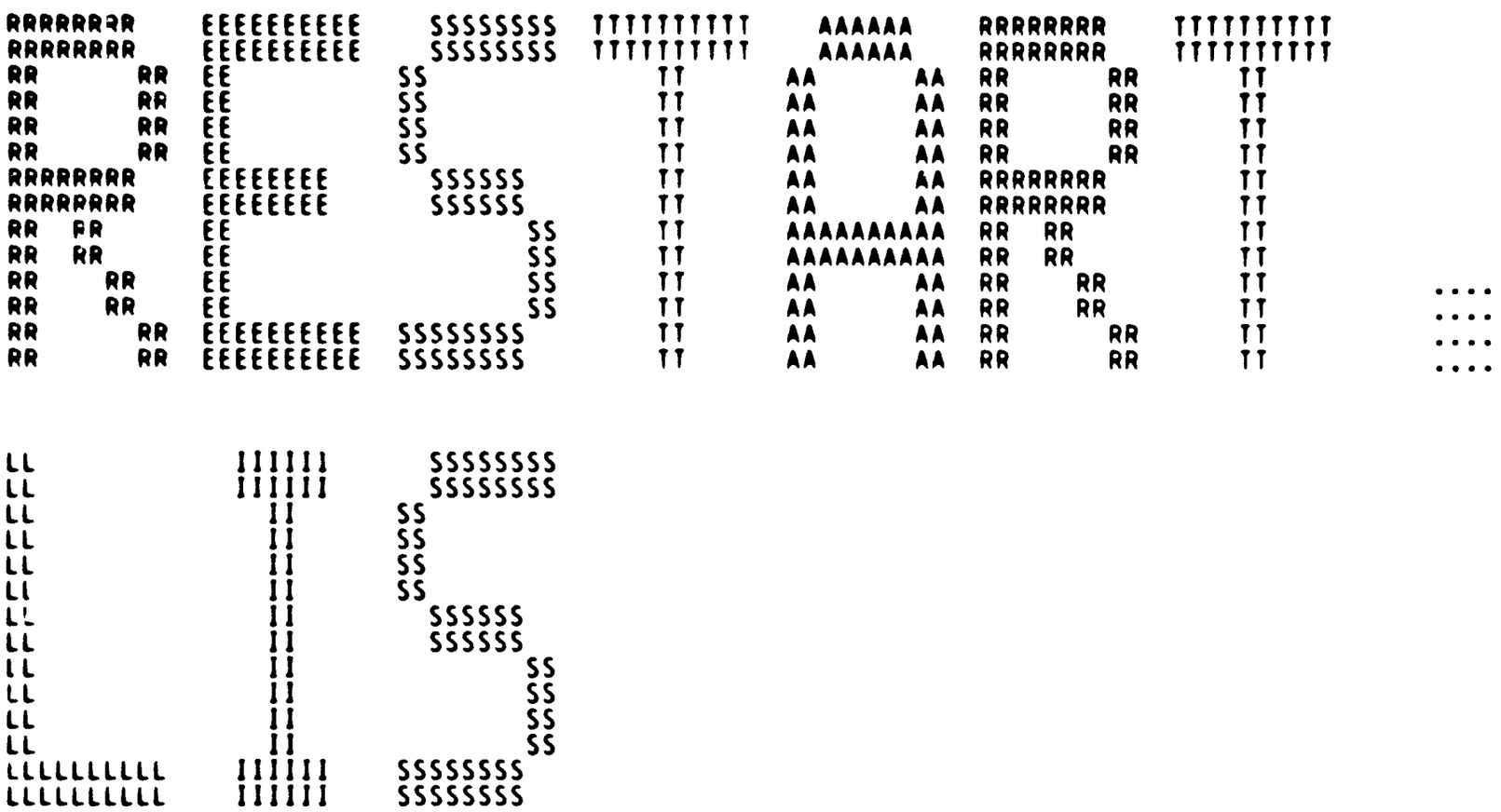
```
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBB         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
```


RESTART
....
....
....
....
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 RESTART  (%TITLE 'Reel Checkpoint and Restart'
;     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 RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
;    25        0025  1 !*  SOFTWARE ON EQUIPMENT 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 checkpoint and restart a save
;    37        0037  1 !       operation from the beginning of a reel.
;    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: 9-May-1981
;    44        0044  1 !
;    45        0045  1 ! MODIFIED BY:
;    46        0046  1 !
;    47        0047  1 !       V03-003 LMP0272         L. Mark Pilant,         6-Jul-1984  8:50
;    48        0048  1 !               Modify BACKUP to always use a full FIB.
;    49        0049  1 !
;    50        0050  1 !       V03-002 LY0458          Larry Yetto             1-FEB-1984 10:20
;    51        0051  1 !               Make restore operation restartable
;    52        0052  1 !
;    53        0053  1 !       V03-001 ACG0313         Andrew C. Goldstein,    12-Feb-1983  16:26
;    54        0054  1 !               Add routine subtitles
;    55        0055  1 !
;    56        0056  1 !       V02-002 MLJ0075         Martin L. Jack, 28-Jan-1982  20:33
;    57        0057  1 !               Use FIB$V_NORECORD.
```

```
;   58          0058  1 !
;   59          0059  1 !       V02-001 MLJ0054         Martin L. Jack, 20-Oct-1981  2:55
;   60          0060  1 !               Implement restart for INPUT_PLACEMENT and INPUT_VBN_LIST.
;   61          0061  1 !               Implement /IGNORE=INTERLOCK.  Move STAACP globals to common.
;   62          0062  1 !               Integrate GET_VM and FREE_VM jacket routines.
;   63          0063  1 !
;   64          0064  1 !--
```

```
;    66          0065  1 REQUIRE 'SRC$:COMMON';
;    67          1171  1 LIBRARY 'SYS$LIBRARY:LIB';
;    68          1172  1 REQUIRE 'LIB$:BACKDEF';
;    69          1622  1
;    70          1623  1
;    71          1624  1 FORWARD ROUTINE
;    72          1625  1         GET_DYN_SPACE:  NOVALUE,        ! Get area of dynamic memory
;    73          1626  1         GET_COPY:       NOVALUE,        ! Copy area to dynamic memory
;    74          1627  1         REEL_CHECKPOINT:NOVALUE,        ! Checkpoint at beginning of reel
;    75          1628  1         RESTORE_COPY:   NOVALUE,        ! Restore copy of dynamic memory
;    76          1629  1         SAVE_RESTART:   NOVALUE,        ! Restart save from beginning of reel
;    77          1630  1         RESTORE_RESTART:NOVALUE;        ! Restart restore from beginning of reel
;    78          1631  1
;    79          1632  1
;    80          1633  1 EXTERNAL ROUTINE
;    81          1634  1         FREE_VM:        NOVALUE,        ! Free virtual memory
;    82          1635  1         GET_VM,                         ! Allocate virtual memory
;    83          1636  1         GET_ZERO_VM,                    ! Allocate and clear virtual memory
;    84          1637  1         CHECKPOINT_M:   NOVALUE,        ! Checkpoint machine state
;    85          1638  1         RESTART_M:      NOVALUE,        ! Restart from checkpointed state
;    86          1639  1         ASSIGN_INPUT_CHANNEL,           ! Assign channel to input volume set
;    87          1640  1         FILE_ERROR:     NOVALUE,        ! Signal file-related error
;    88          1641  1         FREE_BUFFER:    NOVALUE,        ! Free a buffer
;    89          1642  1         WAIT:           NOVALUE,        ! Wait for I/O completion on a buffer
;    90          1643  1         FREE_DIR_DATA:  NOVALUE,        ! Release directory context
;    91          1644  1         INIT_DIR_SCAN:  NOVALUE,        ! Initialize directory context
;    92          1645  1         FIND_NEXT,                      ! Find next file
;    93          1646  1         RESET_DIR_SPEC: NOVALUE;        ! Change selection filespec
;    94          1647  1
;    95          1648  1
;    96          1649  1 EXTERNAL LITERAL
;    97          1650  1         BACKUP$_CONTINUE,
;    98          1651  1         BACKUP$_OPENIN;
;    99          1652  1
;   100          1653  1
;   101          1654  1 G$DEFINE();                            ! Define global area
;   102          1655  1
;   103          1656  1
;   104          1657  1 BUILTIN
;   105          1658  1         CALLG,
;   106          1659  1         INSQUE,
;   107          1660  1         REMQUE;
```

```
;  109          1661  1 %SBTTL 'Checkpoint driver table'
;  110          1662  1 ! Define table to drive checkpointing operation.
;  111          1663  1 !
;  112          1664  1 LITERAL
;  113          1665  1
;  114          1666  1         ! Action codes.
;  115          1667  1         !
;  116          1668  1         EXIT=           0,      ! Exit from operation
;  117          1669  1         COPY=           1,      ! Copy variable
;  118          1670  1         COPYDYN=        2,      ! Copy dynamic area pointed to by variable,
;  119          1671  1                                 ! where length is given by second parameter
;  120          1672  1         SPECIAL_1=      3,      ! Copy dynamic volume information area
;  121          1673  1         SPECIAL_2=      4,      ! Copy index file bitmaps
;  122          1674  1         SPECIAL_3=      5,      ! Copy RMS info for input file
;  123          1675  1         SPECIAL_4=      6,      ! Copy directory positions
;  124          1676  1         SPECIAL_5=      7,      ! Copy FASTSCAN buffer info
;  125          1677  1         SPECIAL_6=      8;      ! Copy file placement blocks
;  126          1678  1
;  127          1679  1
;  128          1680  1 COMPILETIME
;  129          1681  1         VARS_SIZE=      0;      ! Size of area to be allocated
;  130          1682  1
;  131          1683  1
;  132          1684  1 MACRO
;  133          1685  1
;  134          1686  1         ! Macro to generate one table entry:
;  135          1687  1         !       Byte of action code
;  136          1688  1         !       Word of length
;  137          1689  1         !       Word of address relative to GLOBAL_BASE
;  138          1690  1         !
;  139          1691  1         ! Parameters:
;  140          1692  1         !       A = action code
;  141          1693  1         !       B = length, when required
;  142          1694  1         !       C = variable name
;  143          1695  1         !
;  144        M 1696  1         T_[A,B,C]=
;  145        M 1697  1
;  146        M 1698  1             %PRINT('Storage for ', C, ' at offset ', %NUMBER(VARS_SIZE))
;  147        M 1699  1
;  148        M 1700  1             A,
;  149        M 1701  1
;  150        M 1702  1             WORD(
;  151        M 1703  1             %IF A EQL COPY OR A EQL SPECIAL_3 OR A EQL SPECIAL_4
;  152        M 1704  1             %THEN
;  153        M 1705  1                 %IF %NULL(B)
;  154        M 1706  1                 %THEN
;  155        M 1707  1                     %ALLOCATION(C)
;  156        M 1708  1                     %ASSIGN(VARS_SIZE, VARS_SIZE + %ALLOCATION(C))
;  157        M 1709  1                 %ELSE
;  158        M 1710  1                     B
;  159        M 1711  1                     %ASSIGN(VARS_SIZE, VARS_SIZE + B)
;  160        M 1712  1                 %FI
;  161        M 1713  1
;  162        M 1714  1             %ELSE %IF A EQL COPYDYN
;  163        M 1715  1             %THEN
;  164        M 1716  1                 B
;  165        M 1717  1                 %ASSIGN(VARS_SIZE, VARS_SIZE + 8)
```

```
;  166        M 1718  1
;  167        M 1719  1              %ELSE %IF A EQL SPECIAL_1 OR A EQL SPECIAL_2 OR A EQL SPECIAL_5 OR A EQL SPECIAL_6
;  168        M 1720  1              %THEN
;  169        M 1721  1                  0
;  170        M 1722  1                  %ASSIGN(VARS_SIZE, VARS_SIZE + 8)
;  171        M 1723  1
;  172        M 1724  1              %FI %FI %FI
;  173        M 1725  1              )
;  174        M 1726  1
;  175        M 1727  1                , WORD(C - GLOBAL_BASE)
;  176          1728  1              %;
;  177          1729  1
;  178          1730  1 BIND
;  179          1731  1
;  180          1732  1         ! Checkpoint and restart parameter table.  Note well:  COM_I_SETCOUNT
;  181          1733  1         ! and FAST_IMAP_SIZE must retain their existing values until after
;  182          1734  1         ! FAST_IMAP is restored, so they must follow it in the table.  Same
;  183          1735  1         ! for FAST_BUFFER_SIZE vs. FAST_BUFFER.
;  184          1736  1         !
;  185        P 1737  1         CHKPT_TABLE = UPLIT BYTE (T_(
;  186        P 1738  1                 COPY,           2,                      RWSV_VOL_NUMBER,
;  187        P 1739  1                 COPY,           2,                      RWSV_SEG_NUMBER,
;  188        P 1740  1                 COPY,           ,                       RWSV_SAVE_QUAL,
;  189        P 1741  1                 COPY,           ,                       RWSV_IN_SEQ,
;  190        P 1742  1                 COPY,           ,                       RWSV_IN_SEQ_0,
;  191        P 1743  1                 COPY,           ,                       RWSV_IN_VBN,
;  192        P 1744  1                 COPY,           ,                       RWSV_IN_VBN_0,
;  193        P 1745  1                 COPY,           ,                       RWSV_IN_XOR_SEQ,
;  194        P 1746  1                 COPY,           ,                       RWSV_OUT_SEQ,
;  195        P 1747  1                 COPY,           ,                       RWSV_OUT_VBN,
;  196        P 1748  1                 COPY,           ,                       COM_FLAGS,
;  197        P 1749  1                 COPY,           ,                       COM_I_STRUCNAME,
;  198        P 1750  1                 COPY,           ,                       COM_BOFF_COUNT,
;  199        P 1751  1                 COPY,           ,                       FAST_STRUCLEV,
;  200        P 1752  1                 COPY,           INPUT_END-INPUT_BEG,    INPUT_BEG,
;  201        P 1753  1                 COPY,           ,                       INPUT_PROC_LIST,
;  202        P 1754  1                 COPY,           OUTPUT_END-OUTPUT_BEG,  OUTPUT_BEG,
;  203        P 1755  1                 COPY,           ,                       VERIFY_USE_COUNT,
;  204        P 1756  1                 COPY,           ,                       VERIFY_QUAL,
;  205        P 1757  1                 SPECIAL_2,      ,                       FAST_IMAP,
;  206        P 1758  1                 SPECIAL_5,      ,                       FAST_BUFFER,
;  207        P 1759  1                 COPY,           ,                       FAST_BUFFER_SIZE,
;  208        P 1760  1                 SPECIAL_1,      ,                       FAST_IMAP_SIZE,
;  209        P 1761  1                 SPECIAL_1,      ,                       FAST_HDR_OFFSET,
;  210        P 1762  1                 SPECIAL_1,      ,                       FAST_BOOT_LBN,
;  211        P 1763  1                 COPYDYN,        512,                    JOUR_BUFFER,
;  212        P 1764  1                 COPYDYN,        BJL$C_DIR_LEN+1,        JOUR_DIR,
;  213        P 1765  1                 COPY,           ,                       JOUR_EFBLK,
;  214        P 1766  1                 COPY,           ,                       JOUR_FFBYTE,
;  215        P 1767  1                 COPY,           ,                       JOUR_COUNT,
;  216        P 1768  1                 SPECIAL_3,      NAM$C_BLN+NAM$C_MAXRSS, INPUT_FAB,
;  217        P 1769  1                 COPY,           ,                       FAST_RVN,
;  218        P 1770  1                 SPECIAL_4,      D_K_NLEVELS*%UPVAL,     DIR_STACK,
;  219        P 1771  1                 COPY,           ,                       COM_I_SETCOUNT,
;  220        P 1772  1                 COPY,           ,                       INPUT_PLACE_LEN,
;  221        P 1773  1                 SPECIAL_6,      ,                       INPUT_PLACEMENT,
;  222        P 1774  1                 SPECIAL_6,      ,                       INPUT_VBN_LIST
```

```
;  223        L 1775  1                ), LONG(0));
; %PRINT:          Storage for RWSV_VOL_NUMBER at offset 0
; %PRINT:          Storage for RWSV_SEG_NUMBER at offset 2
; %PRINT:          Storage for RWSV_SAVE_QUAL at offset 4
; %PRINT:          Storage for RWSV_IN_SEQ at offset 8
; %PRINT:          Storage for RWSV_IN_SEQ_0 at offset 12
; %PRINT:          Storage for RWSV_IN_VBN at offset 16
; %PRINT:          Storage for RWSV_IN_VBN_0 at offset 20
; %PRINT:          Storage for RWSV_IN_XOR_SEQ at offset 24
; %PRINT:          Storage for RWSV_OUT_SEQ at offset 28
; %PRINT:          Storage for RWSV_OUT_VBN at offset 32
; %PRINT:          Storage for COM_FLAGS at offset 36
; %PRINT:          Storage for COM_I_STRUCNAME at offset 38
; %PRINT:          Storage for COM_BOFF_COUNT at offset 50
; %PRINT:          Storage for FAST_STRUCLEV at offset 51
; %PRINT:          Storage for INPUT_BEG at offset 52
; %PRINT:          Storage for INPUT_PROC_LIST at offset 180
; %PRINT:          Storage for OUTPUT_BEG at offset 184
; %PRINT:          Storage for VERIFY_USE_COUNT at offset 372
; %PRINT:          Storage for VERIFY_QUAL at offset 376
; %PRINT:          Storage for FAST_IMAP at offset 380
; %PRINT:          Storage for FAST_BUFFER at offset 388
; %PRINT:          Storage for FAST_BUFFER_SIZE at offset 396
; %PRINT:          Storage for FAST_IMAP_SIZE at offset 400
; %PRINT:          Storage for FAST_HDR_OFFSET at offset 408
; %PRINT:          Storage for FAST_BOOT_LBN at offset 416
; %PRINT:          Storage for JOUR_BUFFER at offset 424
; %PRINT:          Storage for JOUR_DIR at offset 432
; %PRINT:          Storage for JOUR_EFBLK at offset 440
; %PRINT:          Storage for JOUR_FFBYTE at offset 444
; %PRINT:          Storage for JOUR_COUNT at offset 446
; %PRINT:          Storage for INPUT_FAB at offset 447
; %PRINT:          Storage for FAST_RVN at offset 798
; %PRINT:          Storage for DIR_STACK at offset 799
; %PRINT:          Storage for COM_I_SETCOUNT at offset 835
; %PRINT:          Storage for INPUT_PLACE_LEN at offset 836
; %PRINT:          Storage for INPUT_PLACEMENT at offset 838
; %PRINT:          Storage for INPUT_VBN_LIST at offset 846
```

```
;  225          1776  1 %SBTTL 'GET_DYN_SPACE - allocate dynamic memory'
;  226          1777  1 ROUTINE GET_DYN_SPACE(SRC_LENGTH,SRC_ADDR,DST_DESC): NOVALUE=
;  227          1778  1
;  228          1779  1 !++
;  229          1780  1 !
;  230          1781  1 ! FUNCTIONAL DESCRIPTION:
;  231          1782  1 !       This routine allocates dynamic memory if required.
;  232          1783  1 !
;  233          1784  1 ! INPUT PARAMETERS:
;  234          1785  1 !       SRC_LENGTH      - Length of area to be copied.
;  235          1786  1 !       SRC_ADDR        - Pointer to area to be copied (tested for 0).
;  236          1787  1 !       DST_DESC        - Address of descriptor for dynamic area.
;  237          1788  1 !
;  238          1789  1 ! IMPLICIT INPUTS:
;  239          1790  1 !       NONE
;  240          1791  1 !
;  241          1792  1 ! OUTPUT PARAMETERS:
;  242          1793  1 !       NONE
;  243          1794  1 !
;  244          1795  1 ! IMPLICIT OUTPUTS:
;  245          1796  1 !       NONE
;  246          1797  1 !
;  247          1798  1 ! ROUTINE VALUE:
;  248          1799  1 !       NONE
;  249          1800  1 !
;  250          1801  1 ! SIDE EFFECTS:
;  251          1802  1 !       Dynamic memory allocated.
;  252          1803  1 !
;  253          1804  1 !--
;  254          1805  1
;  255          1806  2 BEGIN
;  256          1807  2 MAP
;  257          1808  2         DST_DESC:       REF VECTOR;     ! Pointer to descriptor
;  258          1809  2
;  259          1810  2
;  260          1811  2 ! Provided it exists, free the old copy of the dynamic area if it is the wrong
;  261          1812  2 ! size or if there is no source data.
;  262          1813  2 !
;  263          1814  2 IF
;  264          1815  2     .DST_DESC[1] NEQ 0 AND
;  265          1816  3     (.SRC_ADDR EQL 0 OR .DST_DESC[0] NEQ .SRC_LENGTH)
;  266          1817  2 THEN
;  267          1818  3     BEGIN
;  268          1819  3     FREE_VM(.DST_DESC[0], .DST_DESC[1]);
;  269          1820  3     DST_DESC[0] = 0;
;  270          1821  3     DST_DESC[1] = 0;
;  271          1822  2     END;
;  272          1823  2
;  273          1824  2
;  274          1825  2 ! If the source area exists, and no dynamic area exists, allocate one.
;  275          1826  2 !
;  276          1827  2 IF .SRC_ADDR NEQ 0 AND .SRC_LENGTH NEQ 0 AND .DST_DESC[1] EQL 0
;  277          1828  2 THEN
;  278          1829  3     BEGIN
;  279          1830  3     DST_DESC[0] = .SRC_LENGTH;
;  280          1831  3     DST_DESC[1] = GET_VM(.SRC_LENGTH);
;  281          1832  2     END;
```

```
;  282          1833  1 END;


                                .TITLE  RESTART Reel Checkpoint and Restart
                                .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

  01  00000 P.AAA:  .BYTE   1
0002  00001         .WORD   2
00EC  00003         .WORD   236
  01  00005         .BYTE   1
0002  00006         .WORD   2
00EE  00008         .WORD   238
  01  0000A         .BYTE   1
0004  0000B         .WORD   4
00F4  0000D         .WORD   244
  01  0000F         .BYTE   1
0004  00010         .WORD   4
0104  00012         .WORD   260
  01  00014         .BYTE   1
0004  00015         .WORD   4
0108  00017         .WORD   264
  01  00019         .BYTE   1
0004  0001A         .WORD   4
0128  0001C         .WORD   296
  01  0001E         .BYTE   1
0004  0001F         .WORD   4
012C  00021         .WORD   300
  01  00023         .BYTE   1
0004  00024         .WORD   4
010C  00026         .WORD   268
  01  00028         .BYTE   1
0004  00029         .WORD   4
0138  0002B         .WORD   312
  01  0002D         .BYTE   1
0004  0002E         .WORD   4
013C  00030         .WORD   316
  01  00032         .BYTE   1
0002  00033         .WORD   2
01C6  00035         .WORD   454
  01  00037         .BYTE   1
000C  00038         .WORD   12
01CC  0003A         .WORD   460
  01  0003C         .BYTE   1
0001  0003D         .WORD   1
01C9  0003F         .WORD   457
  01  00041         .BYTE   1
0001  00042         .WORD   1
020F  00044         .WORD   527
  01  00046         .BYTE   1
0080  00047         .WORD   128
0210  00049         .WORD   528
  01  0004B         .BYTE   1
0004  0004C         .WORD   4
0290  0004E         .WORD   656
  01  00050         .BYTE   1
00BC  00051         .WORD   188
02A8  00053         .WORD   680
  01  00055         .BYTE   1
0004  00056         .WORD   4
0370  00058         .WORD   880
```

```
                01  0005A         .BYTE   1
              0004  0005B         .WORD   4
              0374  0005D         .WORD   884
                04  0005F         .BYTE   4
              0000  00060         .WORD   0
              038C  00062         .WORD   908
                07  00064         .BYTE   7
              0000  00065         .WORD   0
              037C  00067         .WORD   892
                01  00069         .BYTE   1
              0004  0006A         .WORD   4
              0380  0006C         .WORD   896
                03  0006E         .BYTE   3
              0000  0006F         .WORD   0
              0388  00071         .WORD   904
                03  00073         .BYTE   3
              0000  00074         .WORD   0
              0390  00076         .WORD   912
                03  00078         .BYTE   3
              0000  00079         .WORD   0
              0394  0007B         .WORD   916
                02  0007D         .BYTE   2
              0200  0007E         .WORD   512
              0398  00080         .WORD   920
                02  00082         .BYTE   2
              0100  00083         .WORD   256
              039C  00085         .WORD   924
                01  00087         .BYTE   1
              0004  00088         .WORD   4
              03A4  0008A         .WORD   932
                01  0008C         .BYTE   1
              0002  0008D         .WORD   2
              03AC  0008F         .WORD   940
                01  00091         .BYTE   1
              0001  00092         .WORD   1
              03B2  00094         .WORD   946
                05  00096         .BYTE   5
              015F  00097         .WORD   351
              0218  00099         .WORD   536
                01  0009B         .BYTE   1
              0001  0009C         .WORD   1
              0384  0009E         .WORD   900
                06  000A0         .BYTE   6
              0024  000A1         .WORD   36
              052C  000A3         .WORD   1324
                01  000A5         .BYTE   1
              0001  000A6         .WORD   1
              01CA  000A8         .WORD   458
                01  000AA         .BYTE   1
              0002  000AB         .WORD   2
              02A4  000AD         .WORD   676
                08  000AF         .BYTE   8
              0000  000B0         .WORD   0
              0294  000B2         .WORD   660
                08  000B4         .BYTE   8
              0000  000B5         .WORD   0
              029C  000B7         .WORD   668
```

```
                                00000000  000B9          .LONG   0                                       ;

                                                 CHKPT_TABLE=       P.AAA
                                                        .EXTRN  FREE_VM, GET_VM
                                                        .EXTRN  GET_ZERO_VM, CHECKPOINT_M
                                                        .EXTRN  RESTART_M, ASSIGN_INPUT_CHANNEL
                                                        .EXTRN  FILE_ERROR, FREE_BUFFER
                                                        .EXTRN  WAIT, FREE_DIR_DATA
                                                        .EXTRN  INIT_DIR_SCAN, FIND_NEXT
                                                        .EXTRN  RESET_DIR_SPEC, BACKUP$_CONTINUE
                                                        .EXTRN  BACKUP$_OPENIN

                                     0004 00000 GET_DYN_SPACE:
                                                        .WORD   Save R2                                 ; 1777
                              52        0C    AC  D0 00002        MOVL    DST_DESC, R2                    ; 1815
                                        04    A2  D5 00006        TSTL    4(R2)                           ;
                                              17  13 00009        BEQL    2$                              ;
                                        08    AC  D5 0000B        TSTL    SRC_ADDR                        ; 1816
                                              06  13 0000E        BEQL    1$                              ;
                                 04   AC            62  D1 00010        CMPL    (R2), SRC_LENGTH        ;
                                                    0C  13 00014        BEQL    2$                      ;
                                        7E          62  7D 00016 1$:     MOVQ    (R2), -(SP)             ; 1819
                           00000000G    00          02  FB 00019        CALLS   #2, FREE_VM             ;
                                                    62  7C 00020        CLRQ    (R2)                    ; 1820
                                        08    AC  D5 00022 2$:     TSTL    SRC_ADDR                        ; 1827
                                              1C  13 00025        BEQL    3$                              ;
                                        04    AC  D5 00027        TSTL    SRC_LENGTH                      ;
                                              17  13 0002A        BEQL    3$                              ;
                                        04    A2  D5 0002C        TSTL    4(R2)                           ;
                                              12  12 0002F        BNEQ    3$                              ;
                                  62        04    AC  D0 00031        MOVL    SRC_LENGTH, (R2)            ; 1830
                                        04    AC  DD 00035        PUSHL   SRC_LENGTH                      ; 1831
                           00000000G    00          01  FB 00038        CALLS   #1, GET_VM              ;
                                 04     A2          50  D0 0003F        MOVL    R0, 4(R2)               ;
                                                    04  00043 3$:     RET                             ; 1833

; Routine Size:  68 bytes,    Routine Base:  CODE + 00BD
```

```
1 %SBTTL 'GET_COPY - copy memory to allocated space'
1 ROUTINE GET_COPY(SRC_LENGTH,SRC_ADDR,DST_DESC): NOVALUE=
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine allocates dynamic memory if required and copies a
1 !       specified area of memory to it.
1 !
1 ! INPUT PARAMETERS:
1 !       SRC_LENGTH      - Length of area to be copied.
1 !       SRC_ADDR        - Pointer to area to be copied.
1 !       DST_DESC        - Address of descriptor for dynamic area.
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       Dynamic memory allocated.
1 !
1 !--
1
2 BEGIN
2 MAP
2       DST_DESC:       REF VECTOR;     ! Pointer to descriptor
2 BUILTIN
2       AP;
2
2
2 ! Get dynamic space if required.
2 !
2 CALLG(.AP, GET_DYN_SPACE);
2
2
2 ! If the source area exists, copy new data.
2 !
2 IF .DST_DESC[1] NEQ 0
2 THEN
2     CH$MOVE(.SRC_LENGTH, .SRC_ADDR, .DST_DESC[1]);
1 END;
```

```
                              003C 00000 GET_COPY:
                                                 .WORD   Save R2,R3,R4,R5               ; 1835
            B6   AF            6C  FA 00002      CALLG   (AP), GET_DYN_SPACE            ; 1874
```

```
                                50      0C    AC  D0 00006          MOVL    DST_DESC, R0                                ; 1879
                                        04    A0  D5 0000A          TSTL    4(R0)                                       ;
                                              07  13 0000D          BEQL    1$                                          ;
                    04    B0        08    BC        04    AC  28 0000F          MOVC3   SRC_LENGTH, @SRC_ADDR, @4(R0)       ; 1881
                                                        04 00016 1$:      RET                                         ; 1882

; Routine Size:  23 bytes,    Routine Base:  CODE + 0101
```

```
;   334          1883  1 %SBTTL 'REEL_CHECKPOINT - take reel checkpoint'
;   335          1884  1 GLOBAL ROUTINE REEL_CHECKPOINT: NOVALUE=
;   336          1885  1
;   337          1886  1 !++
;   338          1887  1 !
;   339          1888  1 ! FUNCTIONAL DESCRIPTION:
;   340          1889  1 !       This routine takes a checkpoint at the beginning of a reel.
;   341          1890  1 !
;   342          1891  1 ! INPUT PARAMETERS:
;   343          1892  1 !       NONE
;   344          1893  1 !
;   345          1894  1 ! IMPLICIT INPUTS:
;   346          1895  1 !       NONE
;   347          1896  1 !
;   348          1897  1 ! OUTPUT PARAMETERS:
;   349          1898  1 !       NONE
;   350          1899  1 !
;   351          1900  1 ! IMPLICIT OUTPUTS:
;   352          1901  1 !       NONE
;   353          1902  1 !
;   354          1903  1 ! ROUTINE VALUE:
;   355          1904  1 !       NONE
;   356          1905  1 !
;   357          1906  1 ! SIDE EFFECTS:
;   358          1907  1 !       NONE
;   359          1908  1 !
;   360          1909  1 !--
;   361          1910  1
;   362          1911  2 BEGIN
;   363          1912  2 LOCAL
;   364          1913  2         T,                              ! Cursor for CHKPT_TABLE
;   365          1914  2         INPU:           REF BBLOCK,     ! Cursor for input qualifiers area
;   366          1915  2         P:              REF VECTOR;     ! Cursor for dynamic area
;   367          1916  2
;   368          1917  2
;   369          1918  2 ! Determine if a checkpoint at this time is valid.
;   370          1919  2 !
;   371          1920  2 COM_FLAGS[COM_DSBL_RSTRT] = .COM_FLAGS[COM_DSBL_CHKPT];
;   372          1921  2 IF .COM_FLAGS[COM_DSBL_CHKPT] THEN RETURN;
;   373          1922  2
;   374          1923  2
;   375          1924  2 ! Checkpoint the value of QUAL_USE_COUNT in the input qualifiers blocks.
;   376          1925  2 !
;   377          1926  2 INPU = .QUAL[QUAL_INPU_LIST];
;   378          1927  2 WHILE .INPU NEQ 0 DO
;   379          1928  3     BEGIN
;   380          1929  3     INPU[QUAL_USE_CHKPT] = .INPU[QUAL_USE_COUNT];
;   381          1930  3     INPU = .INPU[QUAL_NEXT];
;   382          1931  2     END;
;   383          1932  2
;   384          1933  2
;   385          1934  2 ! Allocate a dynamic area to hold saved variables if none exists.
;   386          1935  2 !
;   387          1936  2 IF .CHKPT_VARS EQL 0 THEN CHKPT_VARS = GET_ZERO_VM(VARS_SIZE);
;   388          1937  2
;   389          1938  2
;   390          1939  2 ! Interpret the table.
```

```
;   391          1940  2 !
;   392          1941  2 T = CHKPT_TABLE;
;   393          1942  2 P = .CHKPT_VARS;
;   394          1943  2 WHILE TRUE DO
;   395          1944  3     BEGIN
;   396          1945  3     LOCAL
;   397          1946  3         A,                              ! Local copy of A byte
;   398          1947  3         B,                              ! Local copy of B byte
;   399          1948  3         C:      REF VECTOR;             ! Local copy of address
;   400          1949  3
;   401          1950  3
;   402          1951  3     ! Establish the three table parameters.
;   403          1952  3     !
;   404          1953  3     A = .(.T)<0,8>;                     T = .T + 1;
;   405          1954  3     B = .(.T)<0,16>;                    T = .T + 2;
;   406          1955  3     C = GLOBAL_BASE + .(.T)<0,16>;      T = .T + 2;
;   407          1956  3
;   408          1957  3
;   409          1958  3     ! Case on the action code to execute the action.
;   410          1959  3     !
;   411          1960  3     CASE .A FROM EXIT TO SPECIAL_6 OF
;   412          1961  3         SET
;   413          1962  3
;   414          1963  3
;   415          1964  3         [EXIT]:
;   416          1965  3             EXITLOOP;
;   417          1966  3
;   418          1967  3
;   419          1968  3         [COPY]:
;   420          1969  3             P = CH$MOVE(.B, .C, .P);    ! Move variable to area
;   421          1970  3
;   422          1971  3
;   423          1972  3         [COPYDYN]:
;   424          1973  4             BEGIN
;   425          1974  4             GET_COPY(.B, ..C, .P);
;   426          1975  4             P = .P + 8;
;   427          1976  3             END;
;   428          1977  3
;   429          1978  3
;   430          1979  3         [SPECIAL_1]:
;   431          1980  4             BEGIN
;   432          1981  4             GET_COPY(.COM_I_SETCOUNT*%UPVAL, ..C, .P);
;   433          1982  4             P = .P + 8;
;   434          1983  3             END;
;   435          1984  3
;   436          1985  3
;   437          1986  3         [SPECIAL_2]:
;   438          1987  4             BEGIN
;   439          1988  4             LOCAL
;   440          1989  4                 Q;
;   441          1990  4
;   442          1991  4             GET_DYN_SPACE(.COM_I_SETCOUNT*2*%UPVAL, ..C, .P);
;   443          1992  4             Q = .P[T];
;   444          1993  4             IF .Q NEQ 0
;   445          1994  4             THEN
;   446          1995  5                 BEGIN
;   447          1996  5                 CH$FILL(0, .COM_I_SETCOUNT*2*%UPVAL, .Q);
```

```
;  448          1997  5                  INCR I FROM 0 TO .COM_I_SETCOUNT-1 DO
;  449          1998  6                      BEGIN
;  450          1999  6                      GET_COPY(.FAST_IMAP_SIZE[.I]*512, .FAST_IMAP[.I], .Q);
;  451          2000  6                      Q = .Q + 8;
;  452          2001  5                      END;
;  453          2002  4                  END;
;  454          2003  4              P = .P + 8;
;  455          2004  3              END;
;  456          2005  3
;  457          2006  3
;  458          2007  3          [SPECIAL_3]:
;  459          2008  4              BEGIN
;  460          2009  4              IF .INPUT_FAB NEQ 0
;  461          2010  4              THEN
;  462          2011  5                  BEGIN
;  463          2012  5                  CH$MOVE(NAM$C_BLN, INPUT_FAB[FC_NAM], .P);
;  464          2013  5                  CH$MOVE(NAM$C_MAXRSS, INPUT_FAB[FC_RSA], .P + NAM$C_BLN);
;  465          2014  4                  END;
;  466          2015  4              P = .P + NAM$C_BLN + NAM$C_MAXRSS;
;  467          2016  3              END;
;  468          2017  3
;  469          2018  3
;  470          2019  3          [SPECIAL_4]:
;  471          2020  4              BEGIN
;  472          2021  4              INCRA D FROM DIR_STACK TO DIR_STACK+D_K_NLEVELS*D_S_ENTRY-D_S_ENTRY BY D_S_ENTRY DO
;  473          2022  5                  BEGIN
;  474          2023  5                  MAP D: REF BBLOCK;
;  475          2024  5                  .P = .D[D_VER];
;  476          2025  5                  P = .P + 4;
;  477          2026  4                  END;
;  478          2027  3              END;
;  479          2028  3
;  480          2029  3
;  481          2030  3          [SPECIAL_5]:
;  482          2031  4              BEGIN
;  483          2032  4              GET_COPY(.FAST_BUFFER_SIZE, ..C, .P);
;  484          2033  4              P = .P + 8;
;  485          2034  3              END;
;  486          2035  3
;  487          2036  3
;  488          2037  3          [SPECIAL_6]:
;  489          2038  4              BEGIN
;  490          2039  4              LOCAL
;  491          2040  4                  Q:      REF BBLOCK,
;  492          2041  4                  R:      REF BBLOCK,
;  493          2042  4                  T:      REF BBLOCK;
;  494          2043  4
;  495          2044  4              IF .C[0] EQL 0 THEN C[0] = C[1] = C[0];
;  496          2045  4              IF .P[0] EQL 0 THEN P[0] = P[1] = P[0];
;  497          2046  4              UNTIL REMQUE(.P[0], T) DO
;  498          2047  5                  BEGIN
;  499          2048  5                  FREE_VM(.T[PLC_SIZE], .T);
;  500          2049  4                  END;
;  501          2050  4              Q = .C[0];
;  502          2051  4              UNTIL .Q EQL C[0] DO
;  503          2052  5                  BEGIN
;  504          2053  5                  R = GET_VM(.Q[PLC_SIZE]);
```

```
;  505     2054  5                    CH$MOVE(.Q[PLC_SIZE], .Q, .R);
;  506     2055  5                    INSQUE(.R, .P[T]);
;  507     2056  5                    Q = .Q[PLC_FLINK];
;  508     2057  4                    END;
;  509     2058  4                P = .P + 8;
;  510     2059  3                END;
;  511     2060  3
;  512     2061  3
;  513     2062  3            TES;
;  514     2063  2        END;
;  515     2064  2
;  516     2065  2
;  517     2066  2 ! Free previous saved machine state if required.
;  518     2067  2 !
;  519     2068  2 IF .CHKPT_STACK NEQ 0
;  520     2069  2 THEN
;  521     2070  2     FREE_VM(.CHKPT_HIGH_SP - .CHKPT_LOW_SP, .CHKPT_STACK);
;  522     2071  2
;  523     2072  2
;  524     2073  2 ! Checkpoint the machine state.  Execution also continues here after a call to
;  525     2074  2 ! SAVE_RESTART.
;  526     2075  2 !
;  527     2076  2 CHECKPOINT_M(.CHKPT_HIGH_SP, CHKPT_STACK, CHKPT_LOW_SP);
;  528     2077  1 END;

                                                  0FFC 00000          .ENTRY  REEL_CHECKPOINT, Save R2,R3,R4,R5,R6,R7,R8,-; 1884
                                                                              R9,R10,R11
                                          5B 00000000'  EF  9E 00002          MOVAB   COM_I_SETCOUNT, R11
        50        FC    AB                01  06  EF 00009          EXTZV   #6, #1, COM_FLAGS, R0                          1920
FC      AB              01                07  50  F0 0000F          INSV    R0, #7, #1, COM_FLAGS
                        01        FC      AB  06  E1 00015          BBC     #6, COM_FLAGS, 1$                              1921
                                              04  00 0001A          RET
                                          50    82  AB  D0 0001B 1$:       MOVL    QUAL, INPU                             1926
                                              0A  13  00 0001F 2$:       BEQL    3$                                     1927
                                  24      A0    20  A0  D0 00021          MOVL    32(INPU), 36(INPU)                     1929
                                          50        60  D0 00026          MOVL    (INPU), INPU                           1930
                                                    F4  11 00029          BRB     2$                                     1927
                                                01FA  CB  D5 0002B 3$:       TSTL    CHKPT_VARS                          1936
                                                    11  12 0002F          BNEQ    4$
                                          7E    0356  8F  3C 00031          MOVZWL  #854, -(SP)
                              00000000G   00        01  FB 00036          CALLS   #1, GET_ZERO_VM
                                  01FA    CB        50  D0 0003D          MOVL    R0, CHKPT_VARS
                                          59    FEA2  CF  9E 00042 4$:       MOVAB   CHKPT_TABLE, T                      1941
                                          57    01FA  CB  D0 00047          MOVL    CHKPT_VARS, P                       1942
                                          52          89  9A 0004C 5$:       MOVZBL  (T)+, A                             1953
                                          51          89  3C 0004F          MOVZWL  (T)+, B                             1954
                                          50    FE36  CB  9E 00052          MOVAB   GLOBAL_BASE, R0                     1955
                                          58          89  3C 00057          MOVZWL  (T)+, C
                                          58          50  C0 0005A          ADDL2   R0, C
                                    08    00          52  CF 0005D          CASEL   A, #0, #8                           1960
                0026          001E          0015          011A     00061 6$:       .WORD   29$-6$,-
                00B3          0098          007A          0034     00069           7$-6$,-
                                                          00C2     00071           8$-6$,-
```

```
                                                                         9$-6$,-                               :
                                                                         11$-6$,-                              :
                                                                         14$-6$,-                              :
                                                                         16$-6$,-                              :
                                                                         19$-6$,-                              :
                                                                         22$-6$                                :
                                   0105  31 00073         BRW            29$                                   : 1965
                 67          68      51  28 00076 7$:     MOVC3          B, (C), (P)                           : 1969
                             57      53  D0 0007A         MOVL           R3, P                                 :
                                     CD  11 0007D         BRB            5$                                    :
                                     57  DD 0007F 8$:     PUSHL          P                                     : 1974
                                     68  DD 00081         PUSHL          (C)                                   :
                                     51  DD 00083         PUSHL          B                                     :
                                     0B  11 00085         BRB            10$                                   :
                                     57  DD 00087 9$:     PUSHL          P                                     : 1981
                                     68  DD 00089         PUSHL          (C)                                   :
                             50      6B  9A 0008B         MOVZBL         COM_I_SETCOUNT, R0                    :
                 7E          50      02  78 0008E         ASHL           #2, R0, -(SP)                         :
                                   0087  31 00092 10$:    BRW            20$                                   :
                                     57  DD 00095 11$:    PUSHL          P                                     : 1991
                                     68  DD 00097         PUSHL          (C)                                   :
                             50      6B  9A 00099         MOVZBL         COM_I_SETCOUNT, R0                    :
                 7E          50      03  78 0009C         ASHL           #3, R0, -(SP)                         :
                      FF00   CF      03  FB 000A0         CALLS          #3, GET_DYN_SPACE                     :
                             56  04  A7  D0 000A5         MOVL           4(P), Q                               : 1992
                                     76  13 000A9         BEQL           21$                                   : 1993
                             50      6B  9A 000AB         MOVZBL         COM_I_SETCOUNT, R0                    : 1996
                             50      08  C4 000AE         MULL2          #8, R0                                :
          50     00          6E      00  2C 000B1         MOVC5          #0, (SP), #0, R0, (Q)                 :
                                     66     000B6                                                              :
                             53      6B  9A 000B7         MOVZBL         COM_I_SETCOUNT, R3                    : 1997
                             52      01  CE 000BA         MNEGL          #1, I                                 :
                                     16  11 000BD         BRB            13$                                   :
                                     56  DD 000BF 12$:    PUSHL          Q                                     : 1999
                             01C2  DB42  DD 000C1         PUSHL          @FAST_IMAP[I]                         :
                 7E   01BE DB42      09  78 000C6         ASHL           #9, @FAST_IMAP_SIZE[I], -(SP)         :
                      FF17   CF      03  FB 000CD         CALLS          #3, GET_COPY                          :
                             56      08  C0 000D2         ADDL2          #8, Q                                 : 2000
                 E6          52      53  F2 000D5 13$:    AOBLSS         R3, I, 12$                            : 1997
                                     46  11 000D9         BRB            21$                                   : 2003
                             56  4E  AB  D0 000DB 14$:    MOVL           INPUT_FAB, R6                         : 2009
                                     11  13 000DF         BEQL           15$                                   :
                 67   0094   C6  0060  8F  28 000E1       MOVC3          #96, 148(R6), (P)                     : 2012
          60     A7   0254   C6  00FF  8F  28 000E9       MOVC3          #255, 596(R6), 96(P)                  : 2013
                             57  015F  C7  9E 000F2 15$:  MOVAB          351(R7), P                            : 2015
                                     7F  11 000F7         BRB            28$                                   : 1960
                             50  0362  CB  9E 000F9 16$:  MOVAB          DIR_STACK, R0                         : 2021
                             51  0582  CB  9E 000FE       MOVAB          DIR_STACK+544, R1                     :
                                     08  11 00103         BRB            18$                                   :
                             87  04  A0  D0 00105 17$:    MOVL           4(D), (P)+                            : 2024
                             50  44  A0  9E 00109         MOVAB          68(R0), D                             : 2021
                             51      50  D1 0010D 18$:    CMPL           D, R1                                 :
                                     F3  1B 00110         BLEQU          17$                                   :
                                     64  11 00112         BRB            28$                                   : 1960
                                     57  DD 00114 19$:    PUSHL          P                                     : 2032
                                     68  DD 00116         PUSHL          (C)                                   :
                             01B6    CB  DD 00118         PUSHL          FAST_BUFFER_SIZE                      :
```

```
                 FEC8   CF                03  FB 0011C 20$:    CALLS   #3, GET_COPY                    ;
                                          52  11 00121 21$:    BRB     27$                             ; 2033
                                          68  D5 00123 22$:    TSTL    (C)                             ; 2044
                                          07  12 00125         BNEQ    23$                             ;
                      04   A8             58  D0 00127         MOVL    C, 4(C)                         ;
                           68             58  D0 0012B         MOVL    C, (C)                          ;
                                          67  D5 0012E 23$:    TSTL    (P)                             ; 2045
                                          07  12 00130         BNEQ    24$                             ;
                      04   A7             57  D0 00132         MOVL    P, 4(P)                         ;
                           67             57  D0 00136         MOVL    P, (P)                          ;
                           52       00    B7  0F 00139 24$:    REMQUE  @0(P), T                        ; 2046
                                          0F  1D 0013D         BVS     25$                             ;
                                          52  DD 0013F         PUSHL   T                               ; 2048
                           7E       09    A2  9A 00141         MOVZBL  9(T), -(SP)                     ;
                 00000000G 00             02  FB 00145         CALLS   #2, FREE_VM                     ;
                                          EB  11 0014C         BRB     24$                             ; 2046
                           56             68  D0 0014E 25$:    MOVL    (C), Q                          ; 2050
                           58             56  D1 00151 26$:    CMPL    Q, C                            ; 2051
                                          1F  13 00154         BEQL    27$                             ;
                           7E       09    A6  9A 00156         MOVZBL  9(Q), -(SP)                     ; 2053
                 00000000G 00             01  FB 0015A         CALLS   #1, GET_VM                      ;
                           5A             50  D0 00161         MOVL    R0, R                           ;
                           50       09    A6  9A 00164         MOVZBL  9(Q), R0                        ; 2054
                6A         66             50  28 00168         MOVC3   R0, (Q), (R)                    ;
                      04   B7             6A  0E 0016C         INSQUE  (R), @4(P)                      ; 2055
                           56             66  D0 00170         MOVL    (Q), Q                          ; 2056
                                          DC  11 00173         BRB     26$                             ; 2051
                           57             08  C0 00175 27$:    ADDL2   #8, P                           ; 2058
                                      FED1    31 00178 28$:    BRW     5$                              ; 1943
                           50   01F6      CB  D0 0017B 29$:    MOVL    CHKPT_STACK, R0                 ; 2068
                                          11  13 00180         BEQL    30$                             ;
                                          50  DD 00182         PUSHL   R0                              ; 2070
                      7E       01EE   CB      01F2  CB  C3 00184         SUBL3   CHKPT_LOW_SP, CHKPT_HIGH_SP, -(SP)  ;
                 00000000G 00                      02  FB 0018C         CALLS   #2, FREE_VM             ;
                                          01F2  CB  9F 00193 30$:    PUSHAB  CHKPT_LOW_SP                ; 2076
                                          01F6  CB  9F 00197         PUSHAB  CHKPT_STACK             ;
                                          01EE  CB  DD 0019B         PUSHL   CHKPT_HIGH_SP           ;
                 00000000G 00                      03  FB 0019F         CALLS   #3, CHECKPOINT_M        ;
                                                   04  001A6         RET                             ; 2077

; Routine Size:  423 bytes,    Routine Base:  CODE + 0118
```

```
;  530          2078  1 %SBTTL 'RESTORE_COPY - restore saved copy of memory'
;  531          2079  1 ROUTINE RESTORE_COPY(SRC_DESC,DST_LENGTH,DST_PTR_ADDR): NOVALUE=
;  532          2080  1
;  533          2081  1 !++
;  534          2082  1 !
;  535          2083  1 ! FUNCTIONAL DESCRIPTION:
;  536          2084  1 !       This routine restores a saved copy of dynamic memory.
;  537          2085  1 !
;  538          2086  1 ! INPUT PARAMETERS:
;  539          2087  1 !       SRC_DESC        - Address of descriptor for dynamic area.
;  540          2088  1 !       DST_LENGTH      - Length of area to be copied.
;  541          2089  1 !       DST_PTR_ADDR    - Pointer to pointer to area to be restored.
;  542          2090  1 !
;  543          2091  1 ! IMPLICIT INPUTS:
;  544          2092  1 !       NONE
;  545          2093  1 !
;  546          2094  1 ! OUTPUT PARAMETERS:
;  547          2095  1 !       NONE
;  548          2096  1 !
;  549          2097  1 ! IMPLICIT OUTPUTS:
;  550          2098  1 !       NONE
;  551          2099  1 !
;  552          2100  1 ! ROUTINE VALUE:
;  553          2101  1 !       NONE
;  554          2102  1 !
;  555          2103  1 ! SIDE EFFECTS:
;  556          2104  1 !       Dynamic memory allocated.
;  557          2105  1 !
;  558          2106  1 !--
;  559          2107  1
;  560          2108  2 BEGIN
;  561          2109  2 MAP
;  562          2110  2         SRC_DESC:       REF VECTOR;     ! Pointer to descriptor
;  563          2111  2
;  564          2112  2
;  565          2113  2 ! Provided it exists, free the old copy of the dynamic area if it is the wrong
;  566          2114  2 ! size or if there is no source data.
;  567          2115  2 !
;  568          2116  2 IF
;  569          2117  2     ..DST_PTR_ADDR NEQ 0 AND
;  570          2118  3     (.SRC_DESC[1] EQL 0 OR .SRC_DESC[0] NEQ .DST_LENGTH)
;  571          2119  2 THEN
;  572          2120  3     BEGIN
;  573          2121  3     FREE_VM(.DST_LENGTH, ..DST_PTR_ADDR);
;  574          2122  3     .DST_PTR_ADDR = 0;
;  575          2123  2     END;
;  576          2124  2
;  577          2125  2
;  578          2126  2 ! If the source area exists, copy new data.
;  579          2127  2 !
;  580          2128  2 IF .SRC_DESC[1] NEQ 0
;  581          2129  2 THEN
;  582          2130  3     BEGIN
;  583          2131  3
;  584          2132  3     ! Allocate a dynamic area if none currently exists.
;  585          2133  3     !
;  586          2134  3     IF ..DST_PTR_ADDR EQL 0 THEN .DST_PTR_ADDR = GET_VM(.SRC_DESC[0]);
```

```
;  587          2135  3
;  588          2136  3
;  589          2137  3      ! Restore the data.
;  590          2138  3      !
;  591          2139  3      CH$MOVE(.SRC_DESC[0], .SRC_DESC[1], ..DST_PTR_ADDR);
;  592          2140  2      END;
;  593          2141  1 END;


                                        003C 00000 RESTORE_COPY:
                                                   .WORD   Save R2,R3,R4,R5                        ; 2079
                            53      0C  AC  D0 00002       MOVL    DST_PTR_ADDR, R3                ; 2117
                                        63  D5 00006       TSTL    (R3)                            ;
                                        1D  13 00008       BEQL    2$                              ;
                            50      04  AC  D0 0000A       MOVL    SRC_DESC, R0                    ; 2118
                                    04  A0  D5 0000E       TSTL    4(R0)                           ;
                                        06  13 00011       BEQL    1$                              ;
                                08  AC      60  D1 00013   CMPL    (R0), DST_LENGTH                ;
                                        0E  13 00017       BEQL    2$                              ;
                                        63  DD 00019 1$:    PUSHL   (R3)                            ; 2121
                                    08  AC  DD 0001B       PUSHL   DST_LENGTH                      ;
                        00000000G   00      02  FB 0001E   CALLS   #2, FREE_VM                     ;
                                        63  D4 00025       CLRL    (R3)                            ; 2122
                            52      04  AC  D0 00027 2$:    MOVL    SRC_DESC, R2                    ; 2128
                                    04  A2  D5 0002B       TSTL    4(R2)                           ;
                                        16  13 0002E       BEQL    4$                              ;
                                        63  D5 00030       TSTL    (R3)                            ; 2134
                                        0C  12 00032       BNEQ    3$                              ;
                                        62  DD 00034       PUSHL   (R2)                            ;
                        00000000G   00      01  FB 00036   CALLS   #1, GET_VM                      ;
                                        63  50  D0 0003D   MOVL    R0, (R3)                        ;
                    00  B3      04  B2      62  28 00040 3$:    MOVC3   (R2), @4(R2), @0(R3)        ; 2139
                                            04  00046 4$:    RET                                     ; 2141

; Routine Size:  71 bytes,    Routine Base:  CODE + 02BF
```

```
;  595          2142  1 %SBTTL 'SAVE_RESTART - restart from last checkpoint'
;  596          2143  1 GLOBAL ROUTINE SAVE_RESTART: NOVALUE=
;  597          2144  1
;  598          2145  1 !++
;  599          2146  1 !
;  600          2147  1 ! FUNCTIONAL DESCRIPTION:
;  601          2148  1 !       This routine restarts from the last checkpoint.
;  602          2149  1 !
;  603          2150  1 ! INPUT PARAMETERS:
;  604          2151  1 !       NONE
;  605          2152  1 !
;  606          2153  1 ! IMPLICIT INPUTS:
;  607          2154  1 !       NONE
;  608          2155  1 !
;  609          2156  1 ! OUTPUT PARAMETERS:
;  610          2157  1 !       NONE
;  611          2158  1 !
;  612          2159  1 ! IMPLICIT OUTPUTS:
;  613          2160  1 !       NONE
;  614          2161  1 !
;  615          2162  1 ! ROUTINE VALUE:
;  616          2163  1 !       NONE
;  617          2164  1 !
;  618          2165  1 ! SIDE EFFECTS:
;  619          2166  1 !       NONE
;  620          2167  1 !
;  621          2168  1 !--
;  622          2169  1
;  623          2170  2 BEGIN
;  624          2171  2 LOCAL
;  625          2172  2         STATUS,                         ! Status variable
;  626          2173  2         T,                              ! Cursor for CHKPT_TABLE
;  627          2174  2         P:              REF BBLOCK,     ! Cursor for dynamic area
;  628          2175  2         SAVE_PROC_LIST: REF BBLOCK,     ! Save for INPUT_PROC_LIST
;  629          2176  2         SAVE_D_VER:     REF VECTOR;     ! Pointer to saved D_VER values
;  630          2177  2
;  631          2178  2 EXTERNAL ROUTINE
;  632          2179  2         STA_DISMOUNT;                   ! Dismount volume via stand-alone ACP
;  633          2180  2
;  634          2181  2
;  635          2182  2 ! Restore the value of QUAL_USE_COUNT in the input qualifiers blocks.
;  636          2183  2 !
;  637          2184  2 P = .QUAL[QUAL_INPU_LIST];
;  638          2185  2 WHILE .P NEQ 0 DO
;  639          2186  3     BEGIN
;  640          2187  3     P[QUAL_USE_COUNT] = .P[QUAL_USE_CHKPT];
;  641          2188  3     P = .P[QUAL_NEXT];
;  642          2189  2     END;
;  643          2190  2
;  644          2191  2 QUAL[QUAL_COMP] = 0;
;  645          2192  2
;  646          2193  2 ! Wait on all pending I/O's.  Reattach all buffers to the free list.
;  647          2194  2 !
;  648          2195  2 UNTIL REMQUE(.INPUT_WAIT[0], P) DO
;  649          2196  3     BEGIN
;  650          2197  3     P[BCB_FAIL_ACT] = 0;
;  651          2198  3     P[BCB_SUCC_ACT] = 0;
```

```
;  652          2199  3     WAIT(.P);
;  653          2200  3     FREE_BUFFER(.P);
;  654          2201  2     END;
;  655          2202  2 UNTIL REMQUE(.REREAD_WAIT[0], P) DO
;  656          2203  3     BEGIN
;  657          2204  3     P[BCB_FAIL_ACT] = 0;
;  658          2205  3     P[BCB_SUCC_ACT] = 0;
;  659          2206  3     WAIT(.P);
;  660          2207  3     FREE_BUFFER(.P);
;  661          2208  2     END;
;  662          2209  2 UNTIL REMQUE(.OUTPUT_WAIT[0], P) DO
;  663          2210  3     BEGIN
;  664          2211  3     P[BCB_FAIL_ACT] = 0;
;  665          2212  3     P[BCB_SUCC_ACT] = 0;
;  666          2213  3     WAIT(.P);
;  667          2214  3     FREE_BUFFER(.P);
;  668          2215  2     END;
;  669          2216  2
;  670          2217  2
;  671          2218  2 ! Deal with buffers that do not have I/O pending.
;  672          2219  2 !
;  673          2220  2 UNTIL REMQUE(.RWSV_HOLD_LIST[0], P) DO
;  674          2221  3     BEGIN
;  675          2222  3     FREE_BUFFER(.P);
;  676          2223  2     END;
;  677          2224  2 IF .RWSV_XOR_BCB NEQ 0
;  678          2225  2 THEN
;  679          2226  3     BEGIN
;  680          2227  3     FREE_BUFFER(.RWSV_XOR_BCB);
;  681          2228  3     RWSV_XOR_BCB = 0;
;  682          2229  2     END;
;  683          2230  2 IF .COMPARE_BCB NEQ 0
;  684          2231  2 THEN
;  685          2232  3     BEGIN
;  686          2233  3     FREE_BUFFER(.COMPARE_BCB);
;  687          2234  3     COMPARE_BCB = 0;
;  688          2235  2     END;
;  689          2236  2 IF .INPUT_BCB NEQ 0
;  690          2237  2 THEN
;  691          2238  3     BEGIN
;  692          2239  3     FREE_BUFFER(.INPUT_BCB);
;  693          2240  3     INPUT_BCB = 0;
;  694          2241  2     END;
;  695          2242  2 IF .OUTPUT_BCB NEQ 0
;  696          2243  2 THEN
;  697          2244  3     BEGIN
;  698          2245  3     FREE_BUFFER(.OUTPUT_BCB);
;  699          2246  3     OUTPUT_BCB = 0;
;  700          2247  2     END;
;  701          2248  2
;  702          2249  2
;  703          2250  2 ! Deassign channels.
;  704          2251  2 ! Close save set if open as a file.
;  705          2252  2 !
;  706          2253  2 IF .QUAL[QUAL_SS_FILE]
;  707          2254  2 THEN
;  708          2255  3     BEGIN
```

```
;  709          2256  3     IF .RWSV_SAVE_FAB NEQ 0 THEN IF .RWSV_SAVE_FAB[FAB$W_IFI] NEQ 0
;  710          2257  3     THEN
;  711          2258  3         $CLOSE(FAB=.RWSV_SAVE_FAB);
;  712          2259  3     END
;  713          2260  3
;  714          2261  3 ! Deassign save set channel.
;  715          2262  3 !
;  716          2263  2 ELSE
;  717          2264  3     BEGIN
;  718          2265  3     IF .RWSV_CHAN NEQ 0
;  719          2266  3     THEN
;  720          2267  4         BEGIN
;  721          2268  4         IF .RWSV_CHAN LSSU 1^16
;  722          2269  4         THEN
;  723          2270  5             BEGIN
;  724          2271  5             $DASSGN(CHAN=.RWSV_CHAN);
;  725          2272  5             RWSV_CHAN = 0;
;  726          2273  5             END
;  727          2274  5
;  728          2275  5 ! Close file and dismount volume if save set is open via stand-alone ACP.
;  729          2276  5 !
;  730          2277  4         ELSE
;  731          2278  5             BEGIN
;  732          2279  5             IF .RWSV_CHAN EQL STA_IN_CHAN
;  733          2280  5             THEN CURRENT_MTL = .INPUT_MTL
;  734          2281  5             ELSE CURRENT_MTL = .OUTPUT_MTL;
;  735        P 2282  5             S$QIOW (CHAN = .RWSV_CHAN,
;  736        P 2283  5                     FUNC = IO$_DEACCESS
;  737          2284  5                     );
;  738          2285  5             STA_DISMOUNT (.RWSV_VOL_NUMBER);
;  739          2286  4             END;
;  740          2287  3         END;
;  741          2288  2     END;
;  742          2289  2 IF .INPUT_CHAN NEQ 0
;  743          2290  2 THEN
;  744          2291  3     BEGIN
;  745        P 2292  3     $QIOW(
;  746        P 2293  3         FUNC=IO$_DEACCESS,
;  747          2294  3         CHAN=.INPUT_CHAN);
;  748          2295  3     $DASSGN(CHAN=.INPUT_CHAN);
;  749          2296  3     INPUT_CHAN = 0;
;  750          2297  2     END;
;  751          2298  2 IF .OUTPUT_CHAN NEQ 0
;  752          2299  2 THEN
;  753          2300  3     BEGIN
;  754        P 2301  3     $QIOW(
;  755        P 2302  3         FUNC=IO$_DEACCESS,
;  756          2303  3         CHAN=.OUTPUT_CHAN);
;  757          2304  3     $DASSGN(CHAN=.OUTPUT_CHAN);
;  758          2305  3     OUTPUT_CHAN = 0;
;  759          2306  2     END;
;  760          2307  2
;  761          2308  2
;  762          2309  2 ! Save globals prior to restoration.
;  763          2310  2 !
;  764          2311  2 SAVE_PROC_LIST = .INPUT_PROC_LIST;
;  765          2312  2
```

```
;  766          2313  2
;  767          2314  2 ! Interpret the table to restore global storage.
;  768          2315  2 !
;  769          2316  2 T = CHKPT_TABLE;
;  770          2317  2 P = .CHKPT_VARS;
;  771          2318  2 WHILE TRUE DO
;  772          2319  3     BEGIN
;  773          2320  3     LOCAL
;  774          2321  3         A,                              ! Local copy of A byte
;  775          2322  3         B,                              ! Local copy of B byte
;  776          2323  3         C:      REF VECTOR;             ! Local copy of address
;  777          2324  3
;  778          2325  3
;  779          2326  3     ! Establish the three table parameters.
;  780          2327  3     !
;  781          2328  3     A = .(.T)<0,8>;                     T = .T + 1;
;  782          2329  3     B = .(.T)<0,16>;                    T = .T + 2;
;  783          2330  3     C = GLOBAL_BASE + .(.T)<0,16>;      T = .T + 2;
;  784          2331  3
;  785          2332  3
;  786          2333  3     ! Case on the action code to execute the action.
;  787          2334  3     !
;  788          2335  3     CASE .A FROM EXIT TO SPECIAL_6 OF
;  789          2336  3         SET
;  790          2337  3
;  791          2338  3
;  792          2339  3         [EXIT]:
;  793          2340  3             EXITLOOP;
;  794          2341  3
;  795          2342  3
;  796          2343  3         [COPY]:
;  797          2344  4             BEGIN
;  798          2345  4             CH$MOVE(.B, .P, .C);        ! Move area to variable
;  799          2346  4             P = .P + .B;                ! Update pointer
;  800          2347  3             END;
;  801          2348  3
;  802          2349  3
;  803          2350  3         [COPYDYN]:
;  804          2351  4             BEGIN
;  805          2352  4             RESTORE_COPY(.P, .B, .C);
;  806          2353  4             P = .P + 8;
;  807          2354  3             END;
;  808          2355  3
;  809          2356  3
;  810          2357  3         [SPECIAL_1]:
;  811          2358  4             BEGIN
;  812          2359  4             RESTORE_COPY(.P, .COM_I_SETCOUNT*%UPVAL, .C);
;  813          2360  4             P = .P + 8;
;  814          2361  3             END;
;  815          2362  3
;  816          2363  3
;  817          2364  3         [SPECIAL_2]:
;  818          2365  4             BEGIN
;  819          2366  4             IF (.P[DSC$A_POINTER] EQL 0 OR .COM_I_SETCOUNT NEQ .P[DSC$W_LENGTH]/(2*%UPVAL)) AND .FAST_IMAP N
;  820          2367  4             THEN
;  821          2368  5                 BEGIN
;  822          2369  5                 INCR I FROM 1 TO .COM_I_SETCOUNT DO
```

```
;  823          2370  6                    BEGIN
;  824          2371  6                    IF .FAST_IMAP[.I-1] NEQ 0
;  825          2372  6                    THEN
;  826          2373  6                        FREE_VM(.FAST_IMAP_SIZE[.I-1] * 512, .FAST_IMAP[.I-1]);
;  827          2374  5                    END;
;  828          2375  5                FREE_VM(.COM_I_SETCOUNT*%UPVAL, .FAST_IMAP);
;  829          2376  5                FAST_IMAP = 0;
;  830          2377  4                END;
;  831          2378  4            IF .P[DSC$A_POINTER] NEQ 0
;  832          2379  4            THEN
;  833          2380  5                BEGIN LOCAL Q;
;  834          2381  5                IF .FAST_IMAP EQL 0
;  835          2382  5                THEN
;  836          2383  6                    BEGIN
;  837          2384  6                    FAST_IMAP = GET_ZERO_VM(.P[DSC$W_LENGTH]/2);
;  838          2385  5                    END;
;  839          2386  5                Q = .P[DSC$A_POINTER];
;  840          2387  5                INCR I FROM 1 TO .COM_I_SETCOUNT DO
;  841          2388  6                    BEGIN
;  842          2389  6                    RESTORE_COPY(
;  843          2390  6                        .Q,
;  844          2391  7                        (IF .FAST_IMAP_SIZE EQL 0
;  845          2392  7                            THEN 0
;  846          2393  6                            ELSE .FAST_IMAP_SIZE[.I-1]),
;  847          2394  6                        FAST_IMAP[.I-1]);
;  848          2395  6                    Q = .Q + 8;
;  849          2396  5                    END;
;  850          2397  4                END;
;  851          2398  4            P = .P + 8;
;  852          2399  3            END;
;  853          2400  3
;  854          2401  3
;  855          2402  3
;  856          2403  3
;  857          2404  3        [SPECIAL_3]:
;  858          2405  4            BEGIN
;  859          2406  4            IF .INPUT_FAB NEQ 0
;  860          2407  4            THEN
;  861          2408  5                BEGIN
;  862          2409  5                CH$MOVE(NAM$C_BLN, .P, INPUT_FAB[FC_NAM]);
;  863          2410  5                CH$MOVE(NAM$C_MAXRSS, .P + NAM$C_BLN, INPUT_FAB[FC_RSA]);
;  864          2411  4                END;
;  865          2412  4            P = .P + NAM$C_BLN + NAM$C_MAXRSS;
;  866          2413  3            END;
;  867          2414  3
;  868          2415  3
;  869          2416  3        [SPECIAL_4]:
;  870          2417  4            BEGIN
;  871          2418  4            SAVE_D_VER = .P;
;  872          2419  4            P = .P + D_K_NLEVELS*%UPVAL;
;  873          2420  3            END;
;  874          2421  3
;  875          2422  3
;  876          2423  3        [SPECIAL_5]:
;  877          2424  4            BEGIN
;  878          2425  4            RESTORE_COPY(.P, .FAST_BUFFER_SIZE, .C);
;  879          2426  4            P = .P + 8;
```

```
;  880          2427  3              END;
;  881          2428  3
;  882          2429  3
;  883          2430  3          [SPECIAL_6]:
;  884          2431  4              BEGIN
;  885          2432  4              LOCAL
;  886          2433  4                  Q:      REF BBLOCK,
;  887          2434  4                  R:      REF BBLOCK,
;  888          2435  4                  T:      REF BBLOCK;
;  889          2436  4              MAP
;  890          2437  4                  P:      REF VECTOR;
;  891          2438  4
;  892          2439  4              IF .C[0] EQL 0 THEN C[0] = C[1] = C[0];
;  893          2440  4              IF .P[0] EQL 0 THEN P[0] = P[1] = P[0];
;  894          2441  4              UNTIL REMQUE(.C[0], T) DO
;  895          2442  5                  BEGIN
;  896          2443  5                  FREE_VM(.T[PLC_SIZE], .T);
;  897          2444  4                  END;
;  898          2445  4              Q = .P[0];
;  899          2446  4              UNTIL .Q EQL P[0] DO
;  900          2447  5                  BEGIN
;  901          2448  5                  R = GET_VM(.Q[PLC_SIZE]);
;  902          2449  5                  CH$MOVE(.Q[PLC_SIZE], .Q, .R);
;  903          2450  5                  INSQUE(.R, .C[T]);
;  904          2451  5                  Q = .Q[PLC_FLINK];
;  905          2452  4                  END;
;  906          2453  4              P = .P + 8;
;  907          2454  3              END;
;  908          2455  3
;  909          2456  3
;  910          2457  3          TES;
;  911          2458  2      END;
;  912          2459  2
;  913          2460  2
;  914          2461  2  ! Reassign channels.
;  915          2462  2  !
;  916          2463  2  IF .INPUT_CHAN NEQ 0
;  917          2464  2  THEN
;  918          2465  3      BEGIN
;  919          2466  3      STATUS = ASSIGN_INPUT_CHANNEL(INPUT_QUAL[QUAL_DEV_DESC], INPUT_CHAN, 0, 0);
;  920          2467  3      IF NOT .STATUS
;  921          2468  3      THEN
;  922          2469  3          FILE_ERROR(
;  923          2470  3              BACKUP$_OPENIN + STS$K_SEVERE,
;  924          2471  3              .INPUT_FAB,
;  925          2472  3              .STATUS);
;  926          2473  2      END;
;  927          2474  2
;  928          2475  2
;  929          2476  2  ! Prune INPUT_PROC_LIST back to its prior state.
;  930          2477  2  !
;  931          2478  2  WHILE .SAVE_PROC_LIST NEQ .INPUT_PROC_LIST DO
;  932          2479  3      BEGIN
;  933          2480  3      LOCAL
;  934          2481  3          T;
;  935          2482  3
;  936          2483  3      T = .SAVE_PROC_LIST;
```

```
;  937          2484  3      SAVE_PROC_LIST = .SAVE_PROC_LIST[REC_NEXT];
;  938          2485  3      FREE_VM(REC_S_ENTRY, .T);
;  939          2486  2      END;
;  940          2487  2
;  941          2488  2
;  942          2489  2  ! Restart file scan.
;  943          2490  2  !
;  944          2491  2  IF NOT .QUAL[QUAL_PHYS] THEN
;  945          2492  2  IF .INPUT_NAM NEQ 0 THEN
;  946          2493  2  IF .INPUT_NAM[NAM$B_RSL] NEQ 0
;  947          2494  2  THEN
;  948          2495  3      BEGIN
;  949          2496  3      IF .INPUT_NAM[NAM$B_DIR] NEQ 2
;  950          2497  3      THEN
;  951          2498  4          BEGIN
;  952          2499  4          LOCAL
;  953          2500  4              RSA:        VECTOR[NAM$C_MAXRSS,BYTE],      ! Copy of filename
;  954          2501  4              DESC:       VECTOR[2];                      ! File name descriptor
;  955          2502  4
;  956          2503  4          FREE_DIR_DATA();
;  957          2504  4          DESC[0] = .INPUT_NAM[NAM$B_RSL];
;  958          2505  4          DESC[1] = RSA;
;  959          2506  4          CH$MOVE(.DESC[0], .INPUT_NAM[NAM$L_RSA], RSA);
;  960          2507  4          INIT_DIR_SCAN(
;  961          2508  4              .INPUT_CHAN,
;  962          2509  4              .INPUT_NAM,
;  963          2510  4              INPUT_QUAL[QUAL_DEV_DESC],
;  964          2511  4              DESC,
;  965          2512  4              %B'10',
;  966          2513  4              .FAST_RVN,
;  967          2514  4              .SAVED_VER);
;  968          2515  4          IF NOT FIND_NEXT()
;  969          2516  4          THEN
;  970          2517  5              BEGIN
;  971          2518  5              INPUT_NAM[NAM$B_RSL] = .DESC[0];
;  972          2519  5              CH$MOVE(.DESC[0], RSA, .INPUT_NAM[NAM$L_RSA]);
;  973          2520  5              COM_FLAGS[COM_FAIL_RSTRT] = TRUE;
;  974          2521  5              CHKPT_STATUS = SS$_NOSUCHFILE;
;  975          2522  4              END;
;  976          2523  4          RESET_DIR_SPEC(
;  977          2524  4              INPUT_QUAL[QUAL_EXP_DESC],
;  978          2525  4              .QUAL[QUAL_IMAG]);
;  979          2526  3          END;
;  980          2527  3
;  981          2528  3
;  982          2529  3      ! If necessary, re-access the file that was accessed at the end of the
;  983          2530  3      ! previous reel.
;  984          2531  3      !
;  985          2532  3      IF
;  986          2533  3          .INPUT_FLAGS[INPUT_OPEN] AND
;  987          2534  3          NOT .QUAL[QUAL_VERI] AND
;  988          2535  3          NOT .COM_FLAGS[COM_FAIL_RSTRT]
;  989          2536  3      THEN
;  990          2537  4          BEGIN
;  991          2538  4          LOCAL
;  992          2539  4              FIB:        BBLOCK[FIB$C_LENGTH],   ! FIB
;  993          2540  4              FIB_DESC:   VECTOR[2],       ! Descriptor for FIB
```

```
;  994          2541  4              IOSB:       VECTOR[4,WORD]; ! I/O status block
;  995          2542  4
;  996          2543  4
;  997          2544  4          CH$FILL (0, FIB$C_LENGTH, FIB);
;  998          2545  4          FIB[FIB$L_ACCTL] = FIB$M_NOWRITE OR FIB$M_NORECORD;
;  999          2546  4          IF .INPUT_FLAGS[INPUT_IGNO_INTE] THEN FIB[FIB$L_ACCTL] = FIB$M_NOLOCK OR FIB$M_NORECORD;
; 1000          2547  4          FIB[FIB$W_FID_NUM] = .INPUT_NAM[NAM$W_FID_NUM];
; 1001          2548  4          FIB[FIB$W_FID_SEQ] = .INPUT_NAM[NAM$W_FID_SEQ];
; 1002          2549  4          FIB[FIB$W_FID_RVN] = .INPUT_NAM[NAM$W_FID_RVN];
; 1003          2550  4          FIB_DESC[0] = FIB$C_LENGTH;
; 1004          2551  4          FIB_DESC[1] = FIB;
; 1005        P 2552  4          STATUS = $QIOW(
; 1006        P 2553  4              FUNC=IO$_ACCESS OR IO$M_ACCESS,
; 1007        P 2554  4              CHAN=.INPUT_CHAN,
; 1008        P 2555  4              IOSB=IOSB,
; 1009          2556  4              P1=FIB_DESC);
; 1010          2557  4          IF .STATUS THEN STATUS = .IOSB[0];
; 1011          2558  4          IF NOT .STATUS
; 1012          2559  4          THEN
; 1013          2560  5              BEGIN
; 1014          2561  5              COM_FLAGS[COM_FAIL_RSTRT] = TRUE;
; 1015          2562  5              CHKPT_STATUS = .STATUS;
; 1016          2563  4              END;
; 1017          2564  3          END;
; 1018          2565  2      END;
; 1019          2566  2
; 1020          2567  2
; 1021          2568  2 ! Restart from the saved machine state.  Execution continues in routine
; 1022          2569  2 ! REEL_CHECKPOINT.
; 1023          2570  2 !
; 1024          2571  2 RESTART_M(.CHKPT_LOW_SP, .CHKPT_HIGH_SP, CHKPT_STACK);
; 1025          2572  1 END;


                                                .EXTRN  STA_DISMOUNT, SYS$CLOSE
                                                .EXTRN  SYS$DASSGN, STA_QIOW
                                                .EXTRN  SYS$QIOW

                                    0FFC 00000          .ENTRY  SAVE_RESTART, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 2143
                                                                R10,R11
                    5E      FEF4    CE   9E 00002               MOVAB   -268(SP), SP                          ;
                    56 00000000'    EF   D0 00007               MOVL    QUAL, P                               ; 2184
                                    0A   13 0000E  1$:          BEQL    2$                                    ; 2185
              20    A6        24    A6   D0 00010               MOVL    36(P), 32(P)                          ; 2187
                    56              66   D0 00015               MOVL    (P), P                                ; 2188
                                    F4   11 00018               BRB     1$                                    ; 2185
 00000000'          EF        80    8F   8A 0001A  2$:          BICB2   #128, QUAL+8                          ; 2191
                    56 00000000'    FF   0F 00022  3$:          REMQUE  @INPUT_WAIT, P                        ; 2195
                                    17   1D 00029               BVS     4$                                    ;
                              20    A6   7C 0002B               CLRQ    32(P)                                 ; 2198
                                    56   DD 0002E               PUSHL   P                                     ; 2199
 00000000G          00              01   FB 00030               CALLS   #1, WAIT                              ;
                                    56   DD 00037               PUSHL   P                                     ; 2200
 00000000G          00              01   FB 00039               CALLS   #1, FREE_BUFFER                       ;
                                    E0   11 00040               BRB     3$                                    ; 2195
                    56 00000000'    FF   0F 00042  4$:          REMQUE  @REREAD_WAIT, P                       ; 2202
```

```
                                    17  1D 00049          BVS     5$                                      ;
                                 20 A6  7C 0004B          CLRQ    32(P)                                   ; 2205
                                    56  DD 0004E          PUSHL   P                                       ; 2206
                00000000G        00 01  FB 00050          CALLS   #1, WAIT                                ;
                                    56  DD 00057          PUSHL   P                                       ; 2207
                00000000G        00 01  FB 00059          CALLS   #1, FREE_BUFFER                         ;
                                    E0  11 00060          BRB     4$                                      ; 2202
                           56 00000000' FF  0F 00062 5$:    REMQUE  @OUTPUT_WAIT, P                         ; 2209
                                    17  1D 00069          BVS     6$                                      ;
                                 20 A6  7C 0006B          CLRQ    32(P)                                   ; 2212
                                    56  DD 0006E          PUSHL   P                                       ; 2213
                00000000G        00 01  FB 00070          CALLS   #1, WAIT                                ;
                                    56  DD 00077          PUSHL   P                                       ; 2214
                00000000G        00 01  FB 00079          CALLS   #1, FREE_BUFFER                         ;
                                    E0  11 00080          BRB     5$                                      ; 2209
                           56 00000000' FF  0F 00082 6$:    REMQUE  @RWSV_HOLD_LIST, P                      ; 2220
                                    0B  1D 00089          BVS     7$                                      ;
                                    56  DD 0008B          PUSHL   P                                       ; 2222
                00000000G        00 01  FB 0008D          CALLS   #1, FREE_BUFFER                         ;
                                    EC  11 00094          BRB     6$                                      ; 2220
                           50 00000000' EF  D0 00096 7$:    MOVL    RWSV_XOR_BCB, R0                        ; 2224
                                    0F  13 0009D          BEQL    8$                                      ;
                                    50  DD 0009F          PUSHL   R0                                      ; 2227
                00000000G        00 01  FB 000A1          CALLS   #1, FREE_BUFFER                         ;
                           00000000' EF  D4 000A8          CLRL    RWSV_XOR_BCB                            ; 2228
                           50 00000000' EF  D0 000AE 8$:    MOVL    COMPARE_BCB, R0                         ; 2230
                                    0F  13 000B5          BEQL    9$                                      ;
                                    50  DD 000B7          PUSHL   R0                                      ; 2233
                00000000G        00 01  FB 000B9          CALLS   #1, FREE_BUFFER                         ;
                           00000000' EF  D4 000C0          CLRL    COMPARE_BCB                             ; 2234
                           50 00000000' EF  D0 000C6 9$:    MOVL    INPUT_BCB, R0                           ; 2236
                                    0F  13 000CD          BEQL    10$                                     ;
                                    50  DD 000CF          PUSHL   R0                                      ; 2239
                00000000G        00 01  FB 000D1          CALLS   #1, FREE_BUFFER                         ;
                           00000000' EF  D4 000D8          CLRL    INPUT_BCB                               ; 2240
                           50 00000000' EF  D0 000DE 10$:   MOVL    OUTPUT_BCB, R0                          ; 2242
                                    0F  13 000E5          BEQL    11$                                     ;
                                    50  DD 000E7          PUSHL   R0                                      ; 2245
                00000000G        00 01  FB 000E9          CALLS   #1, FREE_BUFFER                         ;
                           00000000' EF  D4 000F0          CLRL    OUTPUT_BCB                              ; 2246
             19 00000000' EF        03  E1 000F6 11$:    BBC     #3, QUAL+15, 12$                        ; 2253
                           50 00000000' EF  D0 000FE          MOVL    RWSV_SAVE_FAB, R0                       ; 2256
                                    78  13 00105          BEQL    16$                                     ;
                                 02 A0  B5 00107          TSTW    2(R0)                                   ;
                                    73  13 0010A          BEQL    16$                                     ;
                                    50  DD 0010C          PUSHL   R0                                      ; 2258
                00000000G        00 01  FB 0010E          CALLS   #1, SYS$CLOSE                           ;
                                    68  11 00115          BRB     16$                                     ; 2253
                           52 00000000' EF  D0 00117 12$:   MOVL    RWSV_CHAN, R2                           ; 2265
                                    5F  13 0011E          BEQL    16$                                     ;
                00010000        8F 52  D1 00120          CMPL    R2, #65536                              ; 2268
                                    11  1E 00127          BGEQU   13$                                     ;
                                    52  DD 00129          PUSHL   R2                                      ; 2271
                00000000G        00 01  FB 0012B          CALLS   #1, SYS$DASSGN                          ;
                           00000000' EF  D4 00132          CLRL    RWSV_CHAN                               ; 2272
                                    45  11 00138          BRB     16$                                     ; 2268
                0001FFFF        8F 52  D1 0013A 13$:     CMPL    R2, #131071                             ; 2279
```

```
                                                0D  12 00141          BNEQ    14$
                00000000'  EF 00000000'         EF  D0 00143          MOVL    INPUT_MTL, CURRENT_MTL                          ; 2280
                                                0B  11 0014E          BRB     15$
                00000000'  EF 00000000'         EF  D0 00150 14$:     MOVL    OUTPUT_MTL, CURRENT_MTL                         ; 2281
                                                7E  7C 0015B 15$:     CLRQ    -(SP)                                           ; 2284
                                                7E  7C 0015D          CLRQ    -(SP)
                                                7E  7C 0015F          CLRQ    -(SP)
                                                7E  7C 00161          CLRQ    -(SP)
                           7E                   34  7D 00163          MOVQ    #52, -(SP)
                                                52  DD 00166          PUSHL   R2
                                                7E  D4 00168          CLRL    -(SP)
                00000000G  00                   0C  FB 0016A          CALLS   #12, STA_QIOW
                           7E 00000000'         EF  3C 00171          MOVZWL  RWSV_VOL_NUMBER, -(SP)                          ; 2285
                00000000G  00                   01  FB 00178          CALLS   #1, STA_DISMOUNT
                           50 00000000'         EF  D0 0017F 16$:     MOVL    INPUT_CHAN, R0                                  ; 2289
                                                29  13 00186          BEQL    17$
                                                7E  7C 00188          CLRQ    -(SP)                                           ; 2294
                                                7E  7C 0018A          CLRQ    -(SP)
                                                7E  7C 0018C          CLRQ    -(SP)
                                                7E  7C 0018E          CLRQ    -(SP)
                           7E                   34  7D 00190          MOVQ    #52, -(SP)
                                                50  DD 00193          PUSHL   R0
                                                7E  D4 00195          CLRL    -(SP)
                00000000G  00                   0C  FB 00197          CALLS   #12, SYS$QIOW
                                   00000000'    EF  DD 0019E          PUSHL   INPUT_CHAN                                      ; 2295
                00000000G  00                   01  FB 001A4          CALLS   #1, SYS$DASSGN
                                   00000000'    EF  D4 001AB          CLRL    INPUT_CHAN                                      ; 2296
                           50 00000000'         EF  D0 001B1 17$:     MOVL    OUTPUT_CHAN, R0                                 ; 2298
                                                29  13 001B8          BEQL    18$
                                                7E  7C 001BA          CLRQ    -(SP)                                           ; 2303
                                                7E  7C 001BC          CLRQ    -(SP)
                                                7E  7C 001BE          CLRQ    -(SP)
                                                7E  7C 001C0          CLRQ    -(SP)
                           7E                   34  7D 001C2          MOVQ    #52, -(SP)
                                                50  DD 001C5          PUSHL   R0
                                                7E  D4 001C7          CLRL    -(SP)
                00000000G  00                   0C  FB 001C9          CALLS   #12, SYS$QIOW
                                   00000000'    EF  DD 001D0          PUSHL   OUTPUT_CHAN                                     ; 2304
                00000000G  00                   01  FB 001D6          CALLS   #1, SYS$DASSGN
                                   00000000'    EF  D4 001DD          CLRL    OUTPUT_CHAN                                     ; 2305
                           5B 00000000'         EF  D0 001E3 18$:     MOVL    INPUT_PROC_LIST, SAVE_PROC_LIST                 ; 2311
                           59     FB0C          CF  9E 001EA          MOVAB   CHKPT_TABLE, T                                  ; 2316
                           56 00000000'         EF  D0 001EF          MOVL    CHKPT_VARS, P                                   ; 2317
                           51                   89  9A 001F6 19$:     MOVZBL  (T)+, A                                         ; 2328
                           58                   89  3C 001F9          MOVZWL  (T)+, B                                         ; 2329
                           50 00000000'         EF  9E 001FC          MOVAB   GLOBAL_BASE, R0                                 ; 2330
                           57                   89  3C 00203          MOVZWL  (T)+, C
                           57                   50  C0 00206          ADDL2   R0, C
                        08 00                   51  CF 00209          CASEL   A, #0, #8                                       ; 2335
0024  001E  0015  018D  0020D 20$:     .WORD   49$-20$,-
0124  011C  00FB  0034  00215                  21$-20$,-
                  0135  0021D                  22$-20$,-
                                               23$-20$,-
                                               25$ 20$,-
                                               36$-20$,-
                                               38$-20$,-
                                               39$-20$,-
```

```
                                                             42$-20$                  ;
                                0178  31 0021F      BRW      49$                      ; 2340
              67        66        58  28 00222 21$: MOVC3    B, (P), (C)              ; 2345
                        56        58  C0 00226      ADDL2    B, P                     ; 2346
                                  CB  11 00229      BRB      19$                      ; 2335
                                  57  DD 0022B 22$: PUSHL    C                        ; 2352
                                  58  DD 0022D      PUSHL    B                        ;
                                  0D  11 0022F      BRB      24$                      ;
                                  57  DD 00231 23$: PUSHL    C                        ; 2359
                        50 00000000'  EF  9A 00233  MOVZBL   COM_I_SETCOUNT, R0       ;
              7E        50        02  78 0023A      ASHL     #2, R0, -(SP)            ;
                                00F8  31 0023E 24$: BRW      40$                      ;
                        53     04 A6  D0 00241 25$: MOVL     4(P), R3                 ; 2366
                                  11  13 00245      BEQL     26$                      ;
                        50        66  3C 00247      MOVZWL   (P), R0                  ;
                        50        08  C6 0024A      DIVL2    #8, R0                   ;
  50 00000000'  EF      08        00  ED 0024D      CMPZV    #0, #8, COM_I_SETCOUNT, R0 ;
                                  59  13 00256      BEQL     29$                      ;
                        00000000' EF  D5 00258 26$: TSTL     FAST_IMAP                ;
                                  51  13 0025E      BEQL     29$                      ;
                        54 00000000'  EF  9A 00260  MOVZBL   COM_I_SETCOUNT, R4       ; 2369
                                  52  D4 00267      CLRL     I                        ;
                                  24  11 00269      BRB      28$                      ;
                        50 00000000'FF42  DE 0026B 27$: MOVAL @FAST_IMAP[I], R0       ; 2371
                               FC A0  D5 00273      TSTL     -4(R0)                   ;
                                  17  13 00276      BEQL     28$                      ;
                               FC A0  DD 00278      PUSHL    -4(R0)                   ; 2373
                        50 00000000'FF42  DE 0027B  MOVAL    @FAST_IMAP_SIZE[I], R0   ;
              7E     FC A0        09  78 00283      ASHL     #9, -4(R0), -(SP)        ;
                 00000000G  00    02  FB 00288      CALLS    #2, FREE_VM              ;
              D8        52        54  F3 0028F 28$: AOBLEQ   R4, I, 27$               ; 2369
                        00000000' EF  DD 00293      PUSHL    FAST_IMAP                ; 2375
                        50 00000000'  EF  9A 00299  MOVZBL   COM_I_SETCOUNT, R0       ;
              7E        50        02  78 002A0      ASHL     #2, R0, -(SP)            ;
                 00000000G  00    02  FB 002A4      CALLS    #2, FREE_VM              ;
                        00000000' EF  D4 002AB      CLRL     FAST_IMAP                ; 2376
                                  53  D5 002B1 29$: TSTL     R3                       ; 2378
                                  51  13 002B3      BEQL     35$                      ;
                        00000000' EF  D5 002B5      TSTL     FAST_IMAP                ; 2381
                                  15  12 002BB      BNEQ     30$                      ;
                        50        66  3C 002BD      MOVZWL   (P), R0                  ; 2384
              7E        50        02  C7 002C0      DIVL3    #2, R0, -(SP)            ;
                 00000000G  00    01  FB 002C4      CALLS    #1, GET_ZERO_VM          ;
                 00000000'  EF    50  D0 002CB      MOVL     R0, FAST_IMAP            ;
                        54 00000000'  EF  9A 002D2 30$: MOVZBL COM_I_SETCOUNT, R4     ; 2387
                                  52  D4 002D9      CLRL     I                        ;
                                  25  11 002DB      BRB      34$                      ;
                        00000000'FF42  DF 002DD 31$: PUSHAL @FAST_IMAP[I]             ; 2394
                        6E        04  C2 002E4      SUBL2    #4, (SP)                 ;
                        50 00000000'  EF  D0 002E7  MOVL     FAST_IMAP_SIZE, R0       ; 2391
                                  04  12 002EE      BNEQ     32$                      ;
                                  7E  D4 002F0      CLRL     -(SP)                    ;
                                  04  11 002F2      BRB      33$                      ;
                             FC A042  DD 002F4 32$: PUSHL    -4(R0)[I]                ; 2393
                                  53  DD 002F8 33$: PUSHL    Q                        ; 2394
                   FCBA     CF    03  FB 002FA      CALLS    #3, RESTORE_COPY         ;
                            53    08  C0 002FF      ADDL2    #8, Q                    ; 2395
```

```
                D7              52                  54  F3 00302 34$:   AOBLEQ  R4, I, 31$                        ; 2387
                                                    38  11 00306 35$:   BRB     41$                               ; 2398
                                58 00000000'        EF  D0 00308 36$:   MOVL    INPUT_FAB, R8                     ; 2406
                                                    11  13 0030F       BEQL    37$                               ;
          0094  C8              66      0060        8F  28 00311       MOVC3   #96, (P), 148(R8)                 ; 2409
          0254  C8        60    A6      00FF        8F  28 00319       MOVC3   #255, 96(P), 596(R8)              ; 2410
                                56      015F        C6  9E 00322 37$:   MOVAB   351(R6), P                       ; 2412
                                                    6E  11 00327       BRB     48$                               ; 2335
                                6E                  86  7E 00329 38$:   MOVAQ   (P)+, SAVE_D_VER                  ; 2418
                                56                  1C  C0 0032C       ADDL2   #28, P                            ; 2419
                                                    66  11 0032F       BRB     48$                               ; 2335
                                                    57  DD 00331 39$:   PUSHL   C                                 ; 2425
                                    00000000'       EF  DD 00333       PUSHL   FAST_BUFFER_SIZE                  ;
                                                    56  DD 00339 40$:   PUSHL   P                                 ;
                          FC79  CF                  03  FB 0033B       CALLS   #3, RESTORE_COPY                  ;
                                                    52  11 00340 41$:   BRB     47$                               ; 2426
                                                    67  D5 00342 42$:   TSTL    (C)                               ; 2439
                                                    07  12 00344       BNEQ    43$                               ;
                          04    A7                  57  D0 00346       MOVL    C, 4(C)                           ;
                                67                  57  D0 0034A       MOVL    C, (C)                            ;
                                                    66  D5 0034D 43$:   TSTL    (P)                               ; 2440
                                                    07  12 0034F       BNEQ    44$                               ;
                          04    A6                  56  D0 00351       MOVL    P, 4(P)                           ;
                                66                  56  D0 00355       MOVL    P, (P)                            ;
                                52        00        B7  0F 00358 44$:   REMQUE  @0(C), T                          ; 2441
                                                    0F  1D 0035C       BVS     45$                               ;
                                                    52  DD 0035E       PUSHL   T                                 ; 2443
                                7E        09        A2  9A 00360       MOVZBL  9(T), -(SP)                       ;
                00000000G       00                  02  FB 00364       CALLS   #2, FREE_VM                       ;
                                                    EB  11 0036B       BRB     44$                               ; 2441
                                58                  66  D0 0036D 45$:   MOVL    (P), Q                            ; 2445
                                56                  58  D1 00370 46$:   CMPL    Q, P                              ; 2446
                                                    1F  13 00373       BEQL    47$                               ;
                                7E        09        A8  9A 00375       MOVZBL  9(Q), -(SP)                       ; 2448
                00000000G       00                  01  FB 00379       CALLS   #1, GET_VM                        ;
                                5A                  50  D0 00380       MOVL    R0, R                             ;
                                50        09        A8  9A 00383       MOVZBL  9(Q), R0                          ; 2449
          6A                    68                  50  28 00387       MOVC3   R0, (Q), (R)                      ;
                          04    B7                  6A  0E 0038B       INSQUE  (R), @4(C)                        ; 2450
                                58                  68  D0 0038F       MOVL    (Q), Q                            ; 2451
                                                    DC  11 00392       BRB     46$                               ; 2446
                                56                  08  C0 00394 47$:   ADDL2   #8, P                            ; 2453
                                                FE5C    31 00397 48$:   BRW     19$                               ; 2318
                                    00000000'       EF  D5 0039A 49$:   TSTL    INPUT_CHAN                        ; 2463
                                                    32  13 003A0       BEQL    50$                               ;
                                                    7E  7C 003A2       CLRQ    -(SP)                             ; 2466
                                    00000000'       EF  9F 003A4       PUSHAB  INPUT_CHAN                        ;
          7E    00000000'       EF                  10  C1 003AA       ADDL3   #16, INPUT_QUAL, -(SP)            ;
                00000000G       00                  04  FB 003B2       CALLS   #4, ASSIGN_INPUT_CHANNEL          ;
                                56                  50  D0 003B9       MOVL    R0, STATUS                        ;
                                15                  56  E8 003BC       BLBS    STATUS, 50$                       ; 2467
                                                    56  DD 003BF       PUSHL   STATUS                            ; 2472
                                    00000000'       EF  DD 003C1       PUSHL   INPUT_FAB                         ; 2471
                                    00000000G       8F  DD 003C7       PUSHL   #BACKOPS_OPENIN+4                 ; 2470
                00000000G       00                  03  FB 003CD       CALLS   #3, FILE_ERROR                    ;
                00000000'       EF                  5B  D1 003D4 50$:   CMPL    SAVE_PROC_LIST, INPUT_PROC_LIST   ; 2478
                                                    16  13 003DB       BEQL    51$                               ;
```

```
                                50              5B   D0 003DD          MOVL    SAVE_PROC_LIST, T                       ; 2483
                                5B              6B   D0 003E0          MOVL    (SAVE_PROC_LIST), SAVE_PROC_LIST        ; 2484
                                                50   DD 003E3          PUSHL   T                                       ; 2485
                                7E      030C    8F   3C 003E5          MOVZWL  #780, -(SP)                             ;
                    00000000G   00              02   FB 003EA          CALLS   #2, FREE_VM                             ;
                                                E1   11 003F1          BRB     50$                                     ; 2478
                03 00000000'    EF              05   E1 003F3 51$:     BBC     #5, QUAL+12, 53$                        ; 2491
                                            0132 31   003FB 52$:     BRW     61$                                     ;
                                50 00000000'    EF   D0 003FE 53$:     MOVL    INPUT_NAM, R0                           ; 2492
                                                F4   13 00405          BEQL    52$                                     ;
                                        03      A0   95 00407          TSTB    3(R0)                                   ; 2493
                                                EF   13 0040A          BEQL    52$                                     ;
                                02      3A      A0   91 0040C          CMPB    58(R0), #2                              ; 2496
                                                03   12 00410          BNEQ    54$                                     ;
                                            0089 31   00412          BRW     56$                                     ;
                    00000000G   00              00   FB 00415 54$:     CALLS   #0, FREE_DIR_DATA                       ; 2503
                                57 00000000'    EF   D0 0041C          MOVL    INPUT_NAM, R7                           ; 2504
                        04      AE      03      A7   9A 00423          MOVZBL  3(R7), DESC                             ;
                        08      AE      0C      AE   9E 00428          MOVAB   RSA, DESC+4                             ; 2505
                0C      AE      04      B7      04   AE 28 0042D       MOVC3   DESC, @4(R7), RSA                       ; 2506
                                                6E   DD 00434          PUSHL   SAVE_D_VER                              ; 2514
                                7E 00000000'    EF   9A 00436          MOVZBL  FAST_RVN, -(SP)                         ; 2513
                                                02   DD 0043D          PUSHL   #2                                      ; 2510
                                        10      AE   9F 0043F          PUSHAB  DESC                                    ;
                    7E 00000000'    EF              10   C1 00442          ADDL3   #16, INPUT_QUAL, -(SP)                  ;
                                                57   DD 0044A          PUSHL   R7                                      ;
                                    00000000'   EF   DD 0044C          PUSHL   INPUT_CHAN                              ;
                    00000000G   00              07   FB 00452          CALLS   #7, INIT_DIR_SCAN                       ;
                    00000000G   00              00   FB 00459          CALLS   #0, FIND_NEXT                           ; 2515
                                23              50   E8 00460          BLBS    R0, 55$                                 ;
                                50 00000000'    EF   D0 00463          MOVL    INPUT_NAM, R0                           ; 2518
                                        03      A0   04 AE 90 0046A       MOVB    DESC, 3(R0)                             ;
                        04      B0      0C      AE   04 AE 28 0046F       MOVC3   DESC, RSA, @4(R0)                       ; 2519
                                00000000'   EF              10   88 00476          BISB2   #16, COM_FLAGS                  ; 2520
                                00000000'   EF      0910    8F   3C 0047D          MOVZWL  #2320, CHKPT_STATUS             ; 2521
            7E 00000000'        EF              01          03   EF 00486 55$:     EXTZV   #3, #1, QUAL+10, -(SP)          ; 2525
                                7E 00000000'    EF          08   C1 0048F          ADDL3   #8, INPUT_QUAL, -(SP)           ; 2524
                                00000000G   00              02   FB 00497          CALLS   #2, RESET_DIR_SPEC              ;
                                            03 00000000'    EF   E8 0049E 56$:     BLBS    INPUT_FLAGS, 58$                ; 2533
                                                        0088 31   004A5 57$:     BRW     61$                             ;
                                                00000000'   EF   95 004A8 58$:     TSTB    QUAL+13                         ; 2534
                                                            F5   19 004AE          BLSS    57$                             ;
                                78 00000000'    EF              04   E0 004B0          BBS     #4, COM_FLAGS, 61$              ; 2535
        0040    8F              00              6E          00   2C 004B8          MOVC5   #0, (SP), #0, #64, FIB          ; 2544
                                                        C0  AD   004BF                                                  ;
                                        C0      AD  00200001    8F   D0 004C1          MOVL    #2097153, FIB           ; 2545
                                08 00000000'    EF              04   E1 004C9          BBC     #4, INPUT_FLAGS, 59$            ; 2546
                                        C0      AD  00300000    8F   D0 004D1          MOVL    #3145728, FIB           ;
                                        50  00000000'   EF      D0 004D9 59$:     MOVL    INPUT_NAM, R0                   ; 2547
                                        C4      AD          24  A0   D0 004E0          MOVL    36(R0), FIB+4           ;
                                        C8      AD          28  A0   B0 004E5          MOVW    40(R0), FIB+8           ; 2549
                                        B8      AD          40  8F   9A 004EA          MOVZBL  #64, FIB_DESC           ; 2550
                                        BC      AD          C0  AD   9E 004EF          MOVAB   FIB, FIB_DESC+4         ; 2551
                                                                7E   7C 004F4          CLRQ    -(SP)                   ; 2556
                                                                7E   7C 004F6          CLRQ    -(SP)                   ;
                                                                7E   D4 004F8          CLRL    -(SP)                   ;
                                                        B8      AD   9F 004FA          PUSHAB  FIB_DESC                ;
```

```
                                    7E  7C 004FD          CLRQ    -(SP)                                          ;
                                B0  AD  9F 004FF          PUSHAB  IOSB                                           ;
                             7E    72  8F  9A 00502       MOVZBL  #114, -(SP)                                    ;
                          00000000'  EF  DD 00506          PUSHL   INPUT_CHAN                                     ;
                                    7E  D4 0050C          CLRL    -(SP)                                          ;
                  00000000G   00    0C  FB 0050E          CALLS   #12, SYS$QIOW                                  ;
                              56    50  D0 00515          MOVL    R0, STATUS                                     ;
                              07    56  E9 00518          BLBC    STATUS, 60$                                    ; 2557
                              56  B0  AD  3C 0051B        MOVZWL  IOSB, STATUS                                   ;
                              0E    56  E8 0051F          BLBS    STATUS, 61$                                    ; 2558
                  00000000'   EF    10  88 00522 60$:     BISB2   #16, COM_FLAGS                                 ; 2561
                  00000000'   EF    56  D0 00529          MOVL    STATUS, CHKPT_STATUS                           ; 2562
                          00000000'  EF  9F 00530 61$:     PUSHAB  CHKPT_STACK                                    ; 2571
                          00000000'  EF  DD 00536          PUSHL   CHKPT_HIGH_SP                                  ;
                          00000000'  EF  DD 0053C          PUSHL   CHKPT_LOW_SP                                   ;
                  00000000G   00    03  FB 00542          CALLS   #3, RESTART_M                                  ;
                                        04 00549          RET                                                    ; 2572

; Routine Size:  1354 bytes,    Routine Base:  CODE + 0306
```

```
;  1027          2573  1 %SBTTL 'RESTORE_RESTART - restart restore operation'
;  1028          2574  1 GLOBAL ROUTINE RESTORE_RESTART: NOVALUE=
;  1029          2575  1
;  1030          2576  1 !++
;  1031          2577  1 !
;  1032          2578  1 ! FUNCTIONAL DESCRIPTION:
;  1033          2579  1 !       This routine restarts a restore operation on the current reel
;  1034          2580  1 !
;  1035          2581  1 ! INPUT PARAMETERS:
;  1036          2582  1 !       NONE
;  1037          2583  1 !
;  1038          2584  1 ! IMPLICIT INPUTS:
;  1039          2585  1 !       NONE
;  1040          2586  1 !
;  1041          2587  1 ! OUTPUT PARAMETERS:
;  1042          2588  1 !       NONE
;  1043          2589  1 !
;  1044          2590  1 ! IMPLICIT OUTPUTS:
;  1045          2591  1 !       NONE
;  1046          2592  1 !
;  1047          2593  1 ! ROUTINE VALUE:
;  1048          2594  1 !       NONE
;  1049          2595  1 !
;  1050          2596  1 ! SIDE EFFECTS:
;  1051          2597  1 !       NONE
;  1052          2598  1 !
;  1053          2599  1 !--
;  1054          2600  1
;  1055          2601  2 BEGIN
;  1056          2602  2
;  1057          2603  2 EXTERNAL ROUTINE
;  1058          2604  2     TRY_NEXT_VOLUME ,   ! Set up next volume under handler
;  1059          2605  2     UNLOAD ;            ! Rewind and unload tape
;  1060          2606  2
;  1061          2607  2 UNTIL TRY_NEXT_VOLUME()
;  1062          2608  2 DO UNLOAD();
;  1063          2609  2
;  1064          2610  2 RETURN ;
;  1065          2611  2
;  1066          2612  1 END;


                                                        .EXTRN  TRY_NEXT_VOLUME
                                                        .EXTRN  UNLOAD

                                          0000 00000    .ENTRY  RESTORE_RESTART, Save nothing                   ; 2574
                    00000000G  00       00  FB 00002 1$:     CALLS   #0, TRY_NEXT_VOLUME                     ; 2607
                               09       50  E8 00009         BLBS    R0, 2$                                  ;
                    00000000G  00       00  FB 0000C         CALLS   #0, UNLOAD                              ; 2608
                                        ED  11 00013         BRB     1$                                      ;
                                            04 00015 2$:     RET                                             ; 2612

; Routine Size:  22 bytes,    Routine Base:  CODE + 0850
```

```
; 1068          2613  1 END
; 1069          2614  0 ELUDOM


;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  COMMON                         2124  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  OVR,NOPIC,ALIGN(2)
;  CODE                           2150  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        32         0      1000          00:02.2


;                                       COMMAND QUALIFIERS

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

; Size:          1961 code + 2313 data bytes
; Run Time:         00:48.6
; Elapsed Time:     02:36.4
; Lines/CPU Min:     3230
; Lexemes/CPU-Min: 34523
; Memory Used:  513 pages
; Compilation Complete
```

RESTART
LIS

MAIN
LIS

READSAVE
LIS

LISTJOUR
LIS

OTHERMSG
LIS

MATCH
LIS

SAVE
LIS

RESTORE
LIS

RESTARTM
LIS