```
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
```

```
TTTTTTTTTT    AAAAAA    PPPPPPPP    EEEEEEEEEE  UU      UU  TTTTTTTTTT    IIIIII    LL
TTTTTTTTTT    AAAAAA    PPPPPPPP    EEEEEEEEEE  UU      UU  TTTTTTTTTT    IIIIII    LL
    TT       AA    AA   PP    PP    EE          UU      UU      TT          II      LL
    TT       AA    AA   PP    PP    EE          UU      UU      TT          II      LL
    TT       AA    AA   PP    PP    EE          UU      UU      TT          II      LL
    TT       AA    AA   PP    PP    EE          UU      UU      TT          II      LL
    TT       AA    AA   PPPPPPPP    EEEEEEEE    UU      UU      TT          II      LL
    TT       AA    AA   PPPPPPPP    EEEEEEEE    UU      UU      TT          II      LL
    TT       AAAAAAAAAA PP          EE          UU      UU      TT          II      LL
    TT       AAAAAAAAAA PP          EE          UU      UU      TT          II      LL
    TT       AA    AA   PP          EE          UU      UU      TT          II      LL                ....
    TT       AA    AA   PP          EE          UU      UU      TT          II      LL                ....
    TT       AA    AA   PP          EEEEEEEEEE  UUUUUUUUUU      TT      IIIIII    LLLLLLLLLL        ....
    TT       AA    AA   PP          EEEEEEEEEE  UUUUUUUUUU      TT      IIIIII    LLLLLLLLLL        ....

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

```
;     1          0001  0 MODULE TAPEUTIL (%TITLE 'Magtape Utility Routines'
;     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 !
;    37          0037  1 !       This module contains routines to do the I/O involved in writing
;    38          0038  1 !       save sets.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:
;    41          0041  1 !
;    42          0042  1 !       VAX/VMS user mode
;    43          0043  1 !
;    44          0044  1 !--
;    45          0045  1 !
;    46          0046  1 ! AUTHOR:  Andrew C. Goldstein, CREATION DATE:  17-Sep-1980  21:41
;    47          0047  1 !
;    48          0048  1 ! MODIFIED BY:
;    49          0049  1 !
;    50          0050  1 !       V03-004 LY0457          Larry Yetto             1-FEB-1984 10:18
;    51          0051  1 !               Make restore operation restartable
;    52          0052  1 !
;    53          0053  1 !       V03-003 ACG0315         Andrew C. Goldstein,    10-Feb-1983  23:41
;    54          0054  1 !               Allow label reader to pass bad blocks
;    55          0055  1 !
;    56          0056  1 !       V03-002 ACG0311         Andrew C. Goldstein,    4-Feb-1983  11:23
;    57          0057  1 !               Require YES reply to BACKUP's mount messages.
```

```
;    58          0058  1 !                Also display operator interaction in batch log.
;    59          0059  1 !
;    60          0060  1 !        V03-001 ACG0284         Andrew C. Goldstein,    9-Apr-1982  13:46
;    61          0061  1 !                Complete filtering of EOT status returns
;    62          0062  1 !
;    63          0063  1 !        V02-011 ACG0257         Andrew C. Goldstein,    21-Jan-1982  18:38
;    64          0064  1 !                Support lists of labels
;    65          0065  1 !
;    66          0066  1 !        V02-010 ACG0256         Andrew C. Goldstein,    19-Jan-1982  21:19
;    67          0067  1 !                Add /PROTECTION and /OWNER qualifiers to save sets
;    68          0068  1 !
;    69          0069  1 !        V02-009 MLJ0063         Martin L. Jack, 22-Dec-1981  3:01
;    70          0070  1 !                Use new $PUTMSG action routine parameter.
;    71          0071  1 !
;    72          0072  1 !        V02-008 MLJ0054         Martin L. Jack, 30-Nov-1981  14:03
;    73          0073  1 !                Prompt to OPCOM if and only if SYS$COMMAND is not a terminal.
;    74          0074  1 !
;    75          0075  1 !        V02-007 MLJ0043         Martin L. Jack, 8-Sep-1981  16:47
;    76          0076  1 !                Account for RMS logical device change.
;    77          0077  1 !
;    78          0078  1 !        V02-006 ACG0216         Andrew C. Goldstein,    4-Sep-1981  17:28
;    79          0079  1 !                Make SKIP_RECORD read if only one block
;    80          0080  1 !
;    81          0081  1 !        V02-005 MLJ0036         Martin L. Jack, 29-Aug-1981  16:19
;    82          0082  1 !                Implement operator assisted reel restart.
;    83          0083  1 !
;    84          0084  1 !        V02-004 ACG0209         Andrew C. Goldstein,    10-Jul-1981  13:33
;    85          0085  1 !                Make some errors continuable, fix error messages
;    86          0086  1 !
;    87          0087  1 !        V02-003 MLJ0025         Martin L. Jack, 8-May-1981  14:45
;    88          0088  1 !                Reorganize qualifier database.
;    89          0089  1 !                Make routines non-global if possible.
;    90          0090  1 !
;    91          0091  1 !        V02-002 MLJ0010         Martin L. Jack, 25-Mar-1981  15:30
;    92          0092  1 !                Reorganize global storage.
;    93          0093  1 !
;    94          0094  1 !        V02-001 MLJ0004         Martin L. Jack, 20-Feb-1981  2:10
;    95          0095  1 !                Implement operator assisted tape handling
;    96          0096  1 !
;    97          0097  1 !--
;    98          0098  1
;    99          0099  1
;   100          0100  1 LIBRARY 'SYS$LIBRARY:LIB';
;   101          0101  1 REQUIRE 'SRC$:COMMON';
```

```
;  103          1207  1 FORWARD ROUTINE
;  104          1208  1           OPCOM:              NOVALUE,      ! Communicate with operator
;  105          1209  1           PUTMSG_ACTRTN_B,                  ! $PUTMSG action routine (GET_RESTART)
;  106          1210  1           GET_RESTART:        NOVALUE,      ! Get restart option
;  107          1211  1           SENSE_CHAR,                       ! Read tape characteristics
;  108          1212  1           WRITE_MESSAGE,                    ! Communicate with SYS$COMMAND
;  109          1213  1           MOUNT_MESSAGE     : NOVALUE,      ! issue mount request message
;  110          1214  1           READY_TAPE,                       !
;  111          1215  1           SET_CHAR:           NOVALUE,      ! Set tape characteristics
;  112          1216  1           REWIND:             NOVALUE,      ! Rewind tape
;  113          1217  1           UNLOAD:             NOVALUE,      ! Unload tape
;  114          1218  1           SKIP_TM,                          ! Skip files on tape
;  115          1219  1           SKIP_RECORD,                      ! Skip records on tape
;  116          1220  1           READ_LABEL,                       ! Read label from tape
;  117          1221  1           WRITE_TM:           NOVALUE,      ! Write tape mark on tape
;  118          1222  1           WRITE_LABEL:        NOVALUE,      ! Write label on tape
;  119          1223  1           JULIAN_DATE:        NOVALUE,      ! Julian date conversion
;  120          1224  1           FORMAT_VOLOWNER:NOVALUE,          ! Owner UIC and protection format
;  121          1225  1           MAKE_VOL1:          NOVALUE,      ! Construct VOL1 label
;  122          1226  1           MAKE_HDR1:          NOVALUE,      ! Construct HDR1 label
;  123          1227  1           MAKE_HDR2:          NOVALUE;      ! Construct HDR2 label
;  124          1228  1
;  125          1229  1
;  126          1230  1 EXTERNAL LITERAL
;  127          1231  1           BACKUP$_READYREAD,
;  128          1232  1           BACKUP$_READYWRITE,
;  129          1233  1           BACKUP$_WRITENABLE,
;  130          1234  1           BACKUP$_SPECIFY,
;  131          1235  1           BACKUP$_OPERFAIL,
;  132          1236  1           BACKUP$_OPREPLY,
;  133          1237  1           BACKUP$_ABORT,
;  134          1238  1           BACKUP$_NOTANSI,
;  135          1239  1           BACKUP$_LABELERR,
;  136          1240  1           BACKUP$_POSITERR;
;  137          1241  1
;  138          1242  1
;  139          1243  1 G$DEFINE();                              ! Define global common area
;  140          1244  1
;  141          1245  1
;  142          1246  1 BUILTIN
;  143          1247  1           CALLG;
```

```
;  145          1248  1 %SBTTL 'OPCOM - communicate with OPCOM'
;  146          1249  1 ROUTINE OPCOM (REPLY,MSGID,PARAM): NOVALUE=
;  147          1250  1
;  148          1251  1 !++
;  149          1252  1 !
;  150          1253  1 ! FUNCTIONAL DESCRIPTION:
;  151          1254  1 !       This routine communicates with OPCOM to request tape handling if
;  152          1255  1 !       BACKUP is executing in a batch job.
;  153          1256  1 !
;  154          1257  1 ! INPUT PARAMETERS:
;  155          1258  1 !       REPLY           - Pointer to descriptor to receive reply, or 0
;  156          1259  1 !       MSGID           - Message identification
;  157          1260  1 !       PARAM...        - FAO parameters required by the message
;  158          1261  1 !
;  159          1262  1 ! IMPLICIT INPUTS:
;  160          1263  1 !       NONE
;  161          1264  1 !
;  162          1265  1 ! OUTPUT PARAMETERS:
;  163          1266  1 !       NONE
;  164          1267  1 !
;  165          1268  1 ! IMPLICIT OUTPUTS:
;  166          1269  1 !       NONE
;  167          1270  1 !
;  168          1271  1 ! ROUTINE VALUE:
;  169          1272  1 !       NONE
;  170          1273  1 !
;  171          1274  1 ! SIDE EFFECTS:
;  172          1275  1 !       NONE
;  173          1276  1 !
;  174          1277  1 !--
;  175          1278  1
;  176          1279  2 BEGIN
;  177          1280  2 MAP
;  178          1281  2         REPLY:          REF BBLOCK;             ! Pointer to reply descriptor
;  179          1282  2 LOCAL
;  180          1283  2         CHANNEL,                                ! Channel assigned to mailbox
;  181          1284  2         STATUS,                                 ! Status return
;  182          1285  2         IOSB:           VECTOR[4,WORD],         ! I/O status block
;  183          1286  2         BUFFER_1:       BBLOCK[256],            ! Message buffer
;  184          1287  2         BUFFER_2:       BBLOCK[256],            ! Message buffer
;  185          1288  2         DESC:           VECTOR[2],              ! Descriptor
;  186          1289  2         DESC_2:         VECTOR[2],              ! Descriptor
;  187          1290  2         K,                                      ! Temporary counter
;  188          1291  2         P;                                      ! Temporary
;  189          1292  2
;  190          1293  2
;  191          1294  2 ! Create a mailbox to receive the reply.  The protection on the mailbox is
;  192          1295  2 ! set to system RW, owner RWED for maximum safety.
;  193          1296  2 !
;  194          1297  2 STATUS = $CREMBX(CHAN=CHANNEL, PROMSK=%B'1111111100001100');
;  195          1298  2 IF NOT .STATUS THEN SIGNAL(BACKUP$_OPERFAIL, 0, .STATUS);
;  196          1299  2
;  197          1300  2
;  198          1301  2 ! Get the message text.  The BUFFEROVF or MSGNOTFND status return is
;  199          1302  2 ! considered fatal, since it should not occur.
;  200          1303  2 !
;  201          1304  2 DESC[0] = 256;
```

```
;  202          1305  2 DESC[1] = BUFFER_1;
;  203        P 1306  2 STATUS = $GETMSG(
;  204        P 1307  2     MSGID=.MSGID,
;  205        P 1308  2     MSGLEN=DESC,
;  206          1309  2     BUFADR=DESC);
;  207          1310  2 IF .STATUS NEQ SS$_NORMAL
;  208          1311  2 THEN
;  209          1312  2     SIGNAL(BACKUP$_OPERFAIL, 0, .STATUS);
;  210          1313  2
;  211          1314  2
;  212          1315  2 ! Use FAO to edit the message text.  The gotten message is the control string,
;  213          1316  2 ! and the formatted message is placed in BUFFER_2.
;  214          1317  2 !
;  215          1318  2 DESC_2[0] = 256 - $BYTEOFFSET(OPC$L_MS_TEXT);
;  216          1319  2 DESC_2[1] = BUFFER_2 + $BYTEOFFSET(OPC$L_MS_TEXT);
;  217        P 1320  2 $FAO(
;  218        P 1321  2     CTRSTR=DESC,
;  219        P 1322  2     OUTLEN=DESC_2,
;  220        P 1323  2     OUTBUF=DESC_2,
;  221          1324  2     PRMLST=PARAM);
;  222          1325  2
;  223          1326  2
;  224          1327  2 ! Special case test to remove information following a CRLF from the
;  225          1328  2 ! formatted string.  This allows the same message to be used in interactive
;  226          1329  2 ! and batch processing, but to have the prompt stripped away for batch.
;  227          1330  2 !
;  228          1331  2 P = CH$FIND_CH(.DESC_2[0], .DESC_2[1], %O'015');
;  229          1332  2 IF .P NEQ 0 THEN DESC_2[0] = .P - .DESC_2[1];
;  230          1333  2
;  231          1334  2
;  232          1335  2 ! Initialize OPCOM information at head of buffer.
;  233          1336  2 !
;  234          1337  2 CH$FILL(0, $BYTEOFFSET(OPC$L_MS_TEXT), BUFFER_2);
;  235          1338  2 BUFFER_2[OPC$B_MS_TYPE] = OPC$_RQ_RQST;
;  236          1339  2 BUFFER_2[OPC$B_MS_TARGET] = OPC$M_NM_CENTRL + OPC$M_NM_DEVICE +
;  237          1340  3     (IF .BBLOCK[RQSV_SAVE_FAB[FAB$L_DEV], DEV$V_SQD]
;  238          1341  3         THEN OPC$M_NM_TAPES
;  239          1342  2         ELSE OPC$M_NM_DISKS);
;  240          1343  2
;  241          1344  2
;  242          1345  2 ! Send the message to OPCOM.
;  243          1346  2 !
;  244          1347  2 DESC_2[0] = .DESC_2[0] + $BYTEOFFSET(OPC$L_MS_TEXT);
;  245          1348  2 DESC_2[1] = BUFFER_2;
;  246        P 1349  2 STATUS = $SNDOPR(
;  247        P 1350  2     MSGBUF=DESC_2,
;  248          1351  2     CHAN=.CHANNEL);
;  249          1352  2 IF NOT .STATUS OR .STATUS EQL OPC$_NOPERATOR
;  250          1353  2 THEN
;  251          1354  2     SIGNAL(BACKUP$_OPERFAIL, 0, .STATUS);
;  252          1355  2
;  253          1356  2
;  254          1357  2 WHILE TRUE DO
;  255          1358  3     BEGIN
;  256          1359  3
;  257          1360  3     ! Read the mailbox to get OPCOM's reply.
;  258          1361  3     !
```

```
;  259          P 1362  3      STATUS = $QIOW(
;  260          P 1363  3          FUNC=IO$_READVBLK,
;  261          P 1364  3          CHAN=.CHANNEL,
;  262          P 1365  3          IOSB=IOSB,
;  263          P 1366  3          P1=BUFFER_1,
;  264            1367  3          P2=136);
;  265            1368  3      IF .STATUS THEN STATUS = .IOSB[0];
;  266            1369  3      IF NOT .STATUS THEN SIGNAL(BACKUP$_OPERFAIL, 0, .STATUS);
;  267            1370  3
;  268            1371  3
;  269            1372  3      ! Sanity check the reply buffer.
;  270            1373  3      !
;  271            1374  3      IF
;  272            1375  3          .BUFFER_1[OPC$B_MS_TYPE] NEQ MSG$_OPREPLY OR
;  273            1376  3          .BUFFER_1[OPC$L_MS_RPLYID] NEQ 0
;  274            1377  3      THEN
;  275            1378  3          SIGNAL(BACKUP$_OPERFAIL);
;  276            1379  3
;  277            1380  3
;  278            1381  3      ! Display the reply text, and return it if requested.
;  279            1382  3      ! Remove information following a CRLF from the reply buffer.  This
;  280            1383  3      ! is additional information returned by OPCOM.
;  281            1384  3      !
;  282            1385  3      P = CH$FIND_CH(128, BUFFER_1[OPC$L_MS_TEXT], %O'015');
;  283            1386  3      IF .P EQL 0 THEN P = BUFFER_1[OPC$L_MS_TEXT] + 128;
;  284            1387  3      K = .P - BUFFER_1[OPC$L_MS_TEXT];
;  285            1388  3      IF .K NEQ 0
;  286            1389  3      THEN SIGNAL (BACKUP$_OPREPLY, 2, .K, BUFFER_1[OPC$L_MS_TEXT]);
;  287            1390  3      IF .REPLY NEQ 0
;  288            1391  3      THEN
;  289            1392  4          BEGIN
;  290            1393  4
;  291            1394  4          CH$COPY(
;  292            1395  4              .K, BUFFER_1[OPC$L_MS_TEXT],
;  293            1396  4              %C' ',
;  294            1397  4              .REPLY[DSC$W_LENGTH], .REPLY[DSC$A_POINTER]);
;  295            1398  3          END;
;  296            1399  3
;  297            1400  3
;  298            1401  3      ! Dispatch on the reply type.
;  299            1402  3      !           Request complete:              exit the loop
;  300            1403  3      !           Request pending:               reissue the mailbox read
;  301            1404  3      !           No operator, aborted, etc.:    signal fatal error
;  302            1405  3      !
;  303            1406  3      SELECTONE .BUFFER_1[OPC$W_MS_STATUS] OF
;  304            1407  3          SET
;  305            1408  3
;  306            1409  3          [OPC$_RQSTCMPLTE AND %X'FFFF']:
;  307            1410  3              EXITLOOP;
;  308            1411  3
;  309            1412  3          [OPC$_RQSTPEND AND %X'FFFF']:
;  310            1413  3              0;
;  311            1414  3
;  312            1415  3          [OTHERWISE]:
;  313            1416  3              SIGNAL(
;  314            1417  3                  BACKUP$_OPERFAIL,
;  315            1418  3                  0,
```

```
;  316          1419  3                   (OPC$_RQSTABORT AND %X'FFFF0000') +
;  317          1420  3                       .BUFFER_1[OPC$W_MS_STATUS]);
;  318          1421  3
;  319          1422  3          TES;
;  320          1423  2      END;
;  321          1424  2
;  322          1425  2
;  323          1426  2 ! Delete the mailbox.
;  324          1427  2 !
;  325          1428  2 STATUS = $DASSGN(CHAN=.CHANNEL);
;  326          1429  2 IF NOT .STATUS THEN SIGNAL(BACKUP$_OPERFAIL, 0, .STATUS);
;  327          1430  1 END;
```

```
                                .TITLE  TAPEUTIL Magtape Utility Routines
                                .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

                                        .EXTRN  BACKUP$_READYREAD
                                        .EXTRN  BACKUP$_READYWRITE
                                        .EXTRN  BACKUP$_WRITENABLE
                                        .EXTRN  BACKUP$_SPECIFY
                                        .EXTRN  BACKUP$_OPERFAIL
                                        .EXTRN  BACKUP$_OPREPLY
                                        .EXTRN  BACKUP$_ABORT, BACKUP$_NOTANSI
                                        .EXTRN  BACKUP$_LABELERR
                                        .EXTRN  BACKUP$_POSITERR
                                        .EXTRN  SYS$CREMBX, SYS$GETMSG
                                        .EXTRN  SYS$FAOL, SYS$SNDOPR
                                        .EXTRN  SYS$QIOW, SYS$DASSGN

                                        .PSECT  CODE,NOWRT,2

                    0FFC 00000 OPCOM:   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11         ; 1249
5B 00000000G    8F   D0 00002           MOVL    #BACKUP$_OPERFAIL, R11                      ;
5A 00000000G    00   9E 00009           MOVAB   LIB$SIGNAL, R10                             ;
5E       FDE4   CE   9E 00010           MOVAB   -540(SP), SP                                ;
               7E   7C 00015           CLRQ    -(SP)                                       ; 1297
               7E   FF0C   8F   3C 00017   MOVZWL  #65292, -(SP)                           ;
               7E   7C 0001C           CLRQ    -(SP)                                       ;
                 14   AE   9F 0001E    PUSHAB  CHANNEL                                     ;
               7E   D4 00021           CLRL    -(SP)                                       ;
00000000G      00   07   FB 00023      CALLS   #7, SYS$CREMBX                              ;
               57   50   D0 0002A      MOVL    R0, STATUS                                  ;
               09   57   E8 0002D      BLBS    STATUS, 1$                                  ; 1298
                    57   DD 00030      PUSHL   STATUS                                      ;
                    7E   D4 00032      CLRL    -(SP)                                       ;
                    5B   DD 00034      PUSHL   R11                                         ;
               6A   03   FB 00036      CALLS   #3, LIB$SIGNAL                              ;
          0C   AE   0100   8F   3C 00039 1$: MOVZWL  #256, DESC                            ; 1304
          10   AE   FEF8   CD   9E 0003F   MOVAB   BUFFER_1, DESC+4                       ; 1305
               7E   0F   7D 00045      MOVQ    #15, -(SP)                                  ; 1309
                 14   AE   9F 00048    PUSHAB  DESC                                        ;
                 18   AE   9F 0004B    PUSHAB  DESC                                        ;
                 08   AC   DD 0004E    PUSHL   MSGID                                       ;
00000000G      00   05   FB 00051      CALLS   #5, SYS$GETMSG                              ;
               57   50   D0 00058      MOVL    R0, STATUS                                  ;
               01   57   D1 0005B      CMPL    STATUS, #1                                  ; 1310
                    09   13 0005E      BEQL    2$                                          ;
                    57   DD 00060      PUSHL   STATUS                                      ; 1312
                    7E   D4 00062      CLRL    -(SP)                                       ;
                    5B   DD 00064      PUSHL   R11                                         ;
               6A   03   FB 00066      CALLS   #3, LIB$SIGNAL                              ;
```

```
                                04   AE        F8   8F   9A 00069 2$:     MOVZBL  #248, DESC_2                          ; 1318
                                08   AE        1C   AE   9E 0006E         MOVAB   BUFFER_2+8, DESC_2+4                  ; 1319
                                               0C   AC   9F 00073         PUSHAB  PARAM                                 ; 1324
                                               08   AE   9F 00076         PUSHAB  DESC_2
                                               0C   AE   9F 00079         PUSHAB  DESC_2
                                               18   AE   9F 0007C         PUSHAB  DESC
                           00000000G           00        04   FB 0007F         CALLS   #4, SYS$FAOL
                      08   BE        04   AE             0D   3A 00086         LOCC    #13, DESC_2, @DESC_2+4           ; 1331
                                                         02   12 0008C         BNEQ    3$
                                                         51   D4 0008E         CLRL    R1
                                     58                  51   D0 00090 3$:     MOVL    R1, P
                                                         06   13 00093         BEQL    4$                               ; 1332
                      04   AE                58        08   AE   C3 00095     SUBL3   DESC_2+4, P, DESC_2
                08                 00        6E             00   2C 0009B 4$: MOVC5   #0, (SP), #0, #8, BUFFER_2       ; 1337
                                                    14   AE        000A0
                                04   AE             58   03   90 000A2         MOVB    #3, BUFFER_2                     ; 1338
                                     50 00000000'   EF   D0 000A6         MOVL    RWSV_SAVE_FAB, R0                ; 1340
                           05        40   A0             05   E1 000AD         BBC     #5, 64(R0), 5$
                                     50                  04   D0 000B2         MOVL    #4, R0
                                                         03   11 000B5         BRB     6$
                                     50                  08   D0 000B7 5$:     MOVL    #8, R0
                      15   AE        50                  11   81 000BA 6$:     ADDB3   #17, R0, BUFFER_2+1              ; 1339
                                04   AE                  08   C0 000BF         ADDL2   #8, DESC_2                       ; 1347
                                08   AE        14   AE   9E 000C3         MOVAB   BUFFER_2, DESC_2+4               ; 1348
                                               6E   DD 000C8         PUSHL   CHANNEL                          ; 1351
                                               08   AE   9F 000CA         PUSHAB  DESC_2
                           00000000G           00        02   FB 000CD         CALLS   #2, SYS$SNDOPR
                                     57                  50   D0 000D4         MOVL    R0, STATUS
                                     09                  57   E9 000D7         BLBC    STATUS, 7$                       ; 1352
                           00058061  8F                  57   D1 000DA         CMPL    STATUS, #360545
                                                         09   12 000E1         BNEQ    8$
                                                         57   DD 000E3 7$:     PUSHL   STATUS                           ; 1354
                                                         7E   D4 000E5         CLRL    -(SP)
                                                         5B   DD 000E7         PUSHL   R11
                                6A                       03   FB 000E9         CALLS   #3, LIB$SIGNAL
                                56        04             AC   D0 000EC 8$:     MOVL    REPLY, R6                        ; 1390
                                                         7E   7C 000F0 9$:     CLRQ    -(SP)                            ; 1367
                                                         7E   7C 000F2         CLRQ    -(SP)
                                7E        88             8F   9A 000F4         MOVZBL  #136, -(SP)
                                       FEF8              CD   9F 000F8         PUSHAB  BUFFER_1
                                                         7E   7C 000FC         CLRQ    -(SP)
                                          F8             AD   9F 000FE         PUSHAB  IOSB
                                                         31   DD 00101         PUSHL   #49
                                          28             AE   DD 00103         PUSHL   CHANNEL
                                                         7E   D4 00106         CLRL    -(SP)
                           00000000G      00                  0C   FB 00108         CALLS   #12, SYS$QIOW
                                          57                  50   D0 0010F         MOVL    R0, STATUS
                                          07                  57   E9 00112         BLBC    STATUS, 10$              ; 1368
                                          57             F8   AD   3C 00115         MOVZWL  IOSB, STATUS
                                          09                  57   E8 00119         BLBS    STATUS, 11$              ; 1369
                                                         57   DD 0011C 10$:     PUSHL   STATUS
                                                         7E   D4 0011E         CLRL    -(SP)
                                                         5B   DD 00120         PUSHL   R11
                                6A                       03   FB 00122         CALLS   #3, LIB$SIGNAL
                                09        FEF8           CD   91 00125 11$:     CMPB    BUFFER_1, #9                     ; 1375
                                                         06   12 0012A         BNEQ    12$
                                       FEFC              CD   D5 0012C         TSTL    BUFFER_1+4                       ; 1376
```

```
                                        05  13 00130       BEQL    13$                                      ;
                                        5B  DD 00132 12$:  PUSHL   R11                                      ; 1378
                          6A            01  FB 00134       CALLS   #1, LIB$SIGNAL                           ;
         FF00     CD    0080    8F      0D  3A 00137 13$:  LOCC    #13, #128, BUFFER_1+8                    ; 1385
                                        02  12 0013F       BNEQ    14$                                      ;
                                        51  D4 00141       CLRL    R1                                       ;
                          58            51  D0 00143 14$:  MOVL    R1, P                                    ;
                                        04  12 00146       BNEQ    15$                                      ; 1386
                          58      80    AD  9E 00148       MOVAB   BUFFER_1+136, P                          ;
                          50    FF00    CD  9E 0014C 15$:  MOVAB   BUFFER_1+8, R0                           ; 1387
                   59     58            50  C3 00151       SUBL3   R0, P, K                                 ;
                                        11  13 00155       BEQL    16$                                      ; 1388
                                FF00    CD  9F 00157       PUSHAB  BUFFER_1+8                               ; 1389
                                        59  DD 0015B       PUSHL   K                                        ;
                                        02  DD 0015D       PUSHL   #2                                       ;
                            00000000G   8F  DD 0015F       PUSHL   #BACKUP$_OPREPLY                         ;
                          6A            04  FB 00165       CALLS   #4, LIB$SIGNAL                           ;
                                        56  D5 00168 16$:  TSTL    R6                                       ; 1390
                                        09  13 0016A       BEQL    17$                                      ;
      66           20   FF00      CD    59  2C 0016C       MOVC5   K, BUFFER_1+8, #32, (R6), @4(R6)         ; 1397
                                  04    B6     00173                                                        ;
                          50    FEFA    CD  3C 00175 17$:  MOVZWL  BUFFER_1+2, R0                           ; 1406
                        8029    8F      50  B1 0017A       CMPW    R0, #32809                               ; 1409
                                        17  13 0017F       BEQL    19$                                      ;
                        8021    8F      50  B1 00181       CMPW    R0, #32801                               ; 1412
                                        0D  13 00186       BEQL    18$                                      ;
                            00050000    E0  9F 00188       PUSHAB  327680(R0)                               ; 1419
                                        7E  D4 0018E       CLRL    -(SP)                                    ; 1416
                                        5B  DD 00190       PUSHL   R11                                      ;
                          6A            03  FB 00192       CALLS   #3, LIB$SIGNAL                           ;
                                      FF58  31 00195 18$:  BRW     9$                                       ; 1357
                                        6E  DD 00198 19$:  PUSHL   CHANNEL                                  ; 1428
                   00000000G    00      01  FB 0019A       CALLS   #1, SYS$DASSGN                           ;
                                57      50  D0 001A1       MOVL    R0, STATUS                               ;
                                09      57  E8 001A4       BLBS    STATUS, 20$                              ; 1429
                                        57  DD 001A7       PUSHL   STATUS                                   ;
                                        7E  D4 001A9       CLRL    -(SP)                                    ;
                                        5B  DD 001AB       PUSHL   R11                                      ;
                          6A            03  FB 001AD       CALLS   #3, LIB$SIGNAL                           ;
                                            04 001B0 20$:  RET                                              ; 1430

; Routine Size:  433 bytes,    Routine Base:  CODE + 0000
```

```
;  329          1431  1 %SBTTL 'PUTMSG_ACTRTN_B - handle $PUTMSG output'
;  330          1432  1 ROUTINE PUTMSG_ACTRTN_B (DESC)=
;  331          1433  1
;  332          1434  1 !++
;  333          1435  1 !
;  334          1436  1 ! FUNCTIONAL DESCRIPTION:
;  335          1437  1 !       This routine is an action routine for the $PUTMSG call in routine
;  336          1438  1 !       GET_RESTART.  It takes care of sending the message text to OPCOM
;  337          1439  1 !       if BACKUP is executing in a batch job.  Otherwise, it calls the
;  338          1440  1 !       standard action routine (PUTMSG_ACTRTN), if defined, to put the
;  339          1441  1 !       message out in the standalone environment.  Otherwise, it lets
;  340          1442  1 !       $PUTMSG put the message out.
;  341          1443  1 !
;  342          1444  1 ! INPUT PARAMETERS:
;  343          1445  1 !       DESC            - Descriptor for message.
;  344          1446  1 !
;  345          1447  1 ! IMPLICIT INPUTS:
;  346          1448  1 !       NONE
;  347          1449  1 !
;  348          1450  1 ! OUTPUT PARAMETERS:
;  349          1451  1 !       NONE
;  350          1452  1 !
;  351          1453  1 ! IMPLICIT OUTPUTS:
;  352          1454  1 !       NONE
;  353          1455  1 !
;  354          1456  1 ! ROUTINE VALUE:
;  355          1457  1 !       True if $PUTMSG should put the message out, otherwise false.
;  356          1458  1 !
;  357          1459  1 ! SIDE EFFECTS:
;  358          1460  1 !       NONE
;  359          1461  1 !
;  360          1462  1 !--
;  361          1463  1
;  362          1464  2 BEGIN
;  363          1465  2 MAP
;  364          1466  2         DESC:           REF BBLOCK;     ! Pointer to descriptor
;  365          1467  2 EXTERNAL ROUTINE
;  366          1468  2         PUTMSG_ACTRTN:  WEAK;           ! $PUTMSG action routine for standalone
;  367          1469  2 BUILTIN
;  368          1470  2         AP;
;  369          1471  2
;  370          1472  2
;  371          1473  2 IF NOT .COM_FLAGS[COM_INTERACT]
;  372          1474  2 THEN
;  373          1475  3     BEGIN
;  374          1476  3     LOCAL
;  375          1477  3         BUFFER_2:       BBLOCK[256],            ! OPCOM message buffer
;  376          1478  3         DESC_2:         VECTOR[2],              ! Descriptor for BUFFER_2
;  377          1479  3         STATUS;                                 ! Status variable
;  378          1480  3
;  379          1481  3
;  380          1482  3     ! Initialize OPCOM information at head of buffer and move message text
;  381          1483  3     ! to buffer.
;  382          1484  3     !
;  383          1485  3     CH$FILL(0, $BYTEOFFSET(OPC$L_MS_TEXT), BUFFER_2);
;  384          1486  3     BUFFER_2[OPC$B_MS_TYPE] = OPC$_RQ_RQST;
;  385          1487  3     BUFFER_2[OPC$B_MS_TARGET] = OPC$M_NM_CENTRL + OPC$M_NM_DEVICE +
```

```
;  386          1488  4          (IF .BBLOCK[RWSV_SAVE_FAB[FAB$L_DEV], DEV$V_SQD]
;  387          1489  4              THEN OPC$M_NM_TAPES
;  388          1490  3              ELSE OPC$M_NM_DISKS);
;  389          1491  3      CH$MOVE(.DESC[DSC$W_LENGTH], .DESC[DSC$A_POINTER], BUFFER_2[OPC$L_MS_TEXT]);
;  390          1492  3
;  391          1493  3      ! Send the message to OPCOM.  There is no reply.
;  392          1494  3      !
;  393          1495  3      DESC_2[0] = .DESC[DSC$W_LENGTH] + $BYTEOFFSET(OPC$L_MS_TEXT);
;  394          1496  3      DESC_2[1] = BUFFER_2;
;  395          1497  3      STATUS = $SNDOPR(MSGBUF=DESC_2);
;  396          1498  3      IF NOT .STATUS OR .STATUS EQL OPC$_NOPERATOR
;  397          1499  3      THEN
;  398          1500  3          SIGNAL(BACKUP$_OPERFAIL, 0, .STATUS);
;  399          1501  3
;  400          1502  3
;  401          1503  3      ! Return true to display $PUTMSG output.
;  402          1504  3      !
;  403          1505  3      TRUE
;  404          1506  3      END
;  405          1507  2  ELSE
;  406          1508  3      BEGIN
;  407          1509  3      IF PUTMSG_ACTRTN NEQ 0
;  408          1510  3      THEN
;  409          1511  3          CALLG(.AP, PUTMSG_ACTRTN)
;  410          1512  3      ELSE
;  411          1513  3          TRUE
;  412          1514  3      END
;  413          1515  1  END;


                                                .WEAK   PUTMSG_ACTRTN

                                00FC 00000 PUTMSG_ACTRTN_B:
                                                .WORD   Save R2,R3,R4,R5,R6,R7                  ; 1432
                        57 00000000G    00  9E 00002            MOVAB   PUTMSG_ACTRTN, R7       ;
                        5E      FEF8    CE  9E 00009            MOVAB   -264(SP), SP            ;
                        64 00000000'    EF  E8 0000E            BLBS    COM_FLAGS+1, 4$         ; 1473
08              00      6E              00  2C 00015            MOVC5   #0, (SP), #0, #8, BUFFER_2 ; 1485
                                08      AE      0001A           ;
                            08  AE      03  90 0001C            MOVB    #3, BUFFER_2            ; 1486
                        50 00000000'    EF  D0 00020            MOVL    RWSV_SAVE_FAB, R0       ; 1488
                05      40  A0          05  E1 00027            BBC     #5, 64(R0), 1$          ;
                            50          04  D0 0002C            MOVL    #4, R0                  ;
                                        03  11 0002F            BRB     2$                      ;
                            50          08  D0 00031 1$:        MOVL    #8, R0                  ;
            09  AE          50          11  81 00034 2$:        ADDB3   #17, R0, BUFFER_2+1     ; 1487
                            56      04  AC  D0 00039            MOVL    DESC, R6                ; 1491
            10  AE      04  B6          66  28 0003D            MOVC3   (R6), @4(R6), BUFFER_2+8 ;
                            6E          66  3C 00043            MOVZWL  (R6), DESC_2            ; 1495
                            6E          08  C0 00046            ADDL2   #8, DESC_2              ;
                        04  AE      08  AE  9E 00049            MOVAB   BUFFER_2, DESC_2+4      ; 1496
                                        7E  D4 0004E            CLRL    -(SP)                   ; 1497
                                04  AE  9F 00050               PUSHAB  DESC_2                  ;
                  00000000G     00      02  FB 00053           CALLS   #2, SYS$SNDOPR          ;
                                09      50  E9 0005A           BLBC    STATUS, 3$              ; 1498
                  00058061      8F      50  D1 0005D           CMPL    STATUS, #360545         ;
```

```
                                        1C  12 00064                BNEQ    5$                                      ;
                                        50  DD 00066 3$:            PUSHL   STATUS                                  ; 1500
                                        7E  D4 00068                CLRL    -(SP)                                   ;
                                00000000G   8F  DD 0006A            PUSHL   #BACKUP$_OPERFAIL                       ;
                      00000000G  00         03  FB 00070            CALLS   #3, LIB$SIGNAL                          ;
                                            09  11 00077            BRB     5$                                      ; 1475
                                50          67  9E 00079 4$:        MOVAB   PUTMSG_ACTRTN, R0                       ; 1509
                                            04  13 0007C            BEQL    5$                                      ;
                                67          6C  FA 0007E            CALLG   (AP), PUTMSG_ACTRTN                     ; 1511
                                            04  00081              RET                                             ;
                                50          01  D0 00082 5$:        MOVL    #1, R0                                  ; 1509
                                            04  00085              RET                                             ; 1515

; Routine Size:  134 bytes,    Routine Base:  CODE + 01B1
```

```
;  415          1516  1 %SBTTL 'GET_RESTART - get operator intervention'
;  416          1517  1 GLOBAL ROUTINE GET_RESTART (SIG,MASK): NOVALUE=
;  417          1518  1
;  418          1519  1 !++
;  419          1520  1 !
;  420          1521  1 ! FUNCTIONAL DESCRIPTION:
;  421          1522  1 !       This routine requests a restart option from the interactive user or
;  422          1523  1 !       from the system operator if BACKUP is executing in a batch job.
;  423          1524  1 !
;  424          1525  1 ! INPUT PARAMETERS:
;  425          1526  1 !       SIG             - Signal parameter array
;  426          1527  1 !       MASK            - Bit mask of valid options
;  427          1528  1 !                         %B'001': RESTART is valid
;  428          1529  1 !                         %B'010': CONTINUE is valid
;  429          1530  1 !                                  (QUIT is always valid)
;  430          1531  1 !                         %B'100': restart issued from a restore operation
;  431          1532  1 !
;  432          1533  1 ! IMPLICIT INPUTS:
;  433          1534  1 !       NONE
;  434          1535  1 !
;  435          1536  1 ! OUTPUT PARAMETERS:
;  436          1537  1 !       NONE
;  437          1538  1 !
;  438          1539  1 ! IMPLICIT OUTPUTS:
;  439          1540  1 !       NONE
;  440          1541  1 !
;  441          1542  1 ! ROUTINE VALUE:
;  442          1543  1 !       NONE
;  443          1544  1 !
;  444          1545  1 ! SIDE EFFECTS:
;  445          1546  1 !       If the reply is QUIT:                   Signal the fatal BACKUP$_ABORT
;  446          1547  1 !       If the reply is RESTART
;  447          1548  1 !               to a save operation:            Call SAVE_RESTART
;  448          1549  1 !       If the reply is RESTART
;  449          1550  1 !               to a restore operation:         Call RESTORE_RESTART and return
;  450          1551  1 !       If the reply is CONTINUE:               Return
;  451          1552  1 !
;  452          1553  1 !--
;  453          1554  1
;  454          1555  2 BEGIN
;  455          1556  2 MAP
;  456          1557  2         SIG:            REF BBLOCK;     ! Signal parameters
;  457          1558  2 LOCAL
;  458          1559  2         OPTIONS,                        ! Pointer to ASCIC options list
;  459          1560  2         ANS_BUFFER:     VECTOR[8,BYTE]  ! Buffer for user's response
;  460          1561  2                         VOLATILE;
;  461          1562  2 EXTERNAL ROUTINE
;  462          1563  2         SAVE_RESTART:           NOVALUE,        ! Restart from checkpoint
;  463          1564  2         RESTORE_RESTART:        NOVALUE ;       ! Restart restore operation
;  464          1565  2
;  465          1566  2
;  466          1567  2 ! Issue the original fatal message.
;  467          1568  2 !
;  468          1569  2 BBLOCK[SIG[CHF$L_SIG_NAME], STS$V_SEVERITY] = STS$K_ERROR;
;  469          1570  2 SIG[CHF$L_SIG_ARGS] = .SIG[CHF$L_SIG_ARGS] - 2;
;  470          1571  2 $PUTMSG(MSGVEC=.SIG, ACTRTN=PUTMSG_ACTRTN_B);
;  471          1572  2
```

```
;  472          1573  2
;  473          1574  2 ! Loop until a valid response is received.
;  474          1575  2 !
;  475          1576  2 WHILE TRUE DO
;  476          1577  3     BEGIN
;  477          1578  3     LOCAL
;  478          1579  3         REPLY_DESC:     VECTOR[2];      ! Descriptor for reply
;  479          1580  3
;  480          1581  3
;  481          1582  3     ! Initialize descriptor for answer.
;  482          1583  3     !
;  483          1584  3     REPLY_DESC[0] = 8;
;  484          1585  3     REPLY_DESC[1] = ANS_BUFFER;
;  485          1586  3
;  486          1587  3
;  487          1588  3     CASE (.MASK AND %B'11') FROM %B'01' TO %B'11' OF
;  488          1589  3         SET
;  489          1590  3         [%B'01']:       OPTIONS = UPLIT BYTE (%ASCIC ' or RESTART');
;  490          1591  3         [%B'10']:       OPTIONS = UPLIT BYTE (%ASCIC ' or CONTINUE');
;  491          1592  3         [%B'11']:       OPTIONS = UPLIT BYTE (%ASCIC ', CONTINUE or RESTART');
;  492          1593  3         TES;
;  493          1594  3
;  494          1595  3
;  495          1596  3     ! Issue prompt and receive reply.
;  496          1597  3     !
;  497          1598  3     IF NOT .COM_FLAGS[COM_INTERACT]
;  498          1599  3     THEN
;  499          1600  3         OPCOM(REPLY_DESC,
;  500          1601  3             BACKUP$_SPECIFY,
;  501          1602  3             .OPTIONS)
;  502          1603  3     ELSE
;  503          1604  4         BEGIN
;  504          1605  4         LOCAL
;  505          1606  4             MSG_VECTOR: VECTOR[4];
;  506          1607  4
;  507          1608  4         MSG_VECTOR[0] = 3;
;  508          1609  4         MSG_VECTOR[1] = BACKUP$_SPECIFY;
;  509          1610  4         MSG_VECTOR[2] = 1;
;  510          1611  4         MSG_VECTOR[3] = .OPTIONS;
;  511          1612  4         $PUTMSG(MSGVEC=MSG_VECTOR, ACTRTN=WRITE_MESSAGE, ACTPRM=REPLY_DESC);
;  512          1613  3         END;
;  513          1614  3
;  514          1615  3
;  515          1616  3     ! Analyze reply, after upcasing it.
;  516          1617  3     !
;  517          1618  3     ANS_BUFFER[0] = .ANS_BUFFER[0] AND NOT %O'040';
;  518          1619  3     IF .ANS_BUFFER[0] EQL %C'C' AND .MASK<1,1>
;  519          1620  3     THEN
;  520          1621  3          RETURN;
;  521          1622  3     IF .ANS_BUFFER[0] EQL %C'Q'
;  522          1623  3     THEN
;  523          1624  3         SIGNAL(BACKUP$_ABORT);
;  524          1625  3     IF .ANS_BUFFER[0] EQL %C'R' AND .MASK<0,1>
;  525          1626  3     THEN
;  526          1627  3         IF .MASK<2,1>
;  527          1628  3         THEN
;  528          1629  4             BEGIN
```

```
;  529         1630 4                  RESTORE_RESTART();
;  530         1631 4                  RETURN;
;  531         1632 4                  END
;  533         1633 3              ELSE
;  533         1634 3                  SAVE_RESTART();
;  534         1635 2          END;
;  535         1636 1  END;


            54 52 41 54 53 45 52 20 72 6F 20 0B 00237 P.AAA:  .ASCII  <11>\ or RESTART\                 ;
         45 55 4E 49 54 4E 4F 43 20 72 6F 20 0C 00243 P.AAB:  .ASCII  <12>\ or CONTINUE\                ;
20 72 6F 20 45 55 4E 49 54 4E 4F 43 20 2C 15 00250 P.AAC:  .ASCII  <21>\, CONTINUE or RESTART\      ;
                           54 52 41 54 53 45 52 0025F                                                    ;

                                                         .EXTRN  SAVE_RESTART, RESTORE_RESTART
                                                         .EXTRN  SYS$PUTMSG

                                       003C 00000        .ENTRY  GET_RESTART, Save R2,R3,R4,R5      ; 1517
                        55 00000000G  8F   D0 00002        MOVL    #BACKUP$_SPECIFY, R5               ;
                        54 00000000G  00   9E 00009        MOVAB   SYS$PUTMSG, R4                     ;
                        53      FF37  CF   9E 00010        MOVAB   PUTMSG_ACTRTN_B, R3                ;
                        5E            20   C2 00015        SUBL2   #32, SP                            ;
                        50        04  AC   D0 00018        MOVL    SIG, R0                            ; 1569
      04   A0       03  00            02   F0 0001C        INSV    #2, #0, #3, 4(R0)                  ;
                        60            02   C2 00022        SUBL2   #2, (R0)                           ; 1570
                                      7E   7C 00025        CLRQ    -(SP)                              ; 1571
                                      09   BB 00027        PUSHR   #^M<R0,R3>                         ;
                        64            04   FB 00029        CALLS   #4, SYS$PUTMSG                     ;
                   10   AE            08   D0 0002C 1$:    MOVL    #8, REPLY_DESC                     ; 1584
                   14   AE        18  AE   9E 00030        MOVAB   ANS_BUFFER, REPLY_DESC+4           ; 1585
         50      08  AC   02          00   EF 00035        EXTZV   #0, #2, MASK, R0                   ; 1588
                          02  01          50   CF 0003B        CASEL   R0, #1, #2                         ;
                      0012  000C       0006    0003F 2$:    .WORD   3$-2$,-                            ;
                                                                 4$-2$,-                            ;
                                                                 5$-2$                              ;
                        52        89  AF   9E 00045 3$:    MOVAB   P.AAA, OPTIONS                     ; 1590
                                      0A   11 00049        BRB     6$                                 ;
                        52        8F  AF   9E 0004B 4$:    MOVAB   P.AAB, OPTIONS                     ; 1591
                                      04   11 0004F        BRB     6$                                 ;
                        52        96  AF   9E 00051 5$:    MOVAB   P.AAC, OPTIONS                     ; 1592
                        0E 00000000'  EF   E8 00055 6$:    BLBS    COM_FLAGS+1, 7$                    ; 1598
                                      52   DD 0005C        PUSHL   OPTIONS                            ; 1602
                                      55   DD 0005E        PUSHL   R5                                 ; 1600
                                  18  AE   9F 00060        PUSHAB  REPLY_DESC                         ;
                 FE4F   C3            03   FB 00063        CALLS   #3, OPCOM                          ;
                                      1E   11 00068        BRB     8$                                 ;
                        6E            03   D0 0006A 7$:    MOVL    #3, MSG_VECTOR                     ; 1608
                   04   AE            55   D0 0006D        MOVL    R5, MSG_VECTOR+4                   ; 1609
                   08   AE            01   D0 00071        MOVL    #1, MSG_VECTOR+8                   ; 1610
                   0C   AE            52   D0 00075        MOVL    OPTIONS, MSG_VECTOR+12             ; 1611
                                  10  AE   9F 00079        PUSHAB  REPLY_DESC                         ; 1612
                                      7E   D4 0007C        CLRL    -(SP)                              ;
                               0000V  CF   9F 0007E        PUSHAB  WRITE_MESSAGE                      ;
                                  0C  AE   9F 00082        PUSHAB  MSG_VECTOR                         ;
                        64            04   FB 00085        CALLS   #4, SYS$PUTMSG                     ;
                   18   AE            20   8A 00088 8$:    BICB2   #32, ANS_BUFFER                    ; 1618
```

```
                         43   8F        18   AE  91 0008C          CMPB    ANS_BUFFER, #67                          ; 1619
                                        05   12  00091            BNEQ    9$                                       ;
                   36    08   AC             01  E0 00093        BBS     #1, MASK, 13$                            ;
                         51   8F        18   AE  91 00098 9$:     CMPB    ANS_BUFFER, #81                          ; 1622
                                        0D   12  0009D            BNEQ    10$                                      ;
                                00000000G   8F  DD 0009F        PUSHL   #BACKUP$_ABORT                           ; 1624
                C0000000G   00              01  FB 000A5        CALLS   #1, LIB$SIGNAL                           ;
                         52   8F        18   AE  91 000AC 10$:    CMPB    ANS_BUFFER, #82                          ; 1625
                                        18   12  000B1            BNEQ    12$                                      ;
                              14        08   AC  E9 000B3        BLBC    MASK, 12$                                ;
                   08    08   AC             02  E1 000B7        BBC     #2, MASK, 11$                            ; 1627
                00000000G   00              00  FB 000BC        CALLS   #0, RESTORE_RESTART                      ; 1630
                                            04  000C3            RET                                              ; 1629
                00000000G   00          00      FB 000C4 11$:    CALLS   #0, SAVE_RESTART                         ; 1634
                                    FF5E    31  000CB 12$:    BRW     1$                                       ; 1576
                                            04  000CE 13$:    RET                                              ; 1636

; Routine Size:  207 bytes,    Routine Base:  CODE + 0266
```

```
;  537          1637  1 %SBTTL 'SENSE_CHAR - sense tape charactertistics'
;  538          1638  1 GLOBAL ROUTINE SENSE_CHAR =
;  539          1639  1
;  540          1640  1 !++
;  541          1641  1 !
;  542          1642  1 ! FUNCTIONAL DESCRIPTION:
;  543          1643  1 !
;  544          1644  1 !       This routine reads the tape characteristics.
;  545          1645  1 !
;  546          1646  1 ! CALLING SEQUENCE:
;  547          1647  1 !       SENSE_CHAR ()
;  548          1648  1 !
;  549          1649  1 ! INPUT PARAMETERS:
;  550          1650  1 !       NONE
;  551          1651  1 !
;  552          1652  1 ! IMPLICIT INPUTS:
;  553          1653  1 !       NONE
;  554          1654  1 !
;  555          1655  1 ! OUTPUT PARAMETERS:
;  556          1656  1 !       NONE
;  557          1657  1 !
;  558          1658  1 ! IMPLICIT OUTPUTS:
;  559          1659  1 !       NONE
;  560          1660  1 !
;  561          1661  1 ! ROUTINE VALUE :
;  562          1662  1 !       tape device characteristics longword
;  563          1663  1 !
;  564          1664  1 ! SIDE EFFECTS:
;  565          1665  1 !       NONE
;  566          1666  1 !
;  567          1667  1 !--
;  568          1668  1
;  569          1669  2 BEGIN
;  570          1670  2 LOCAL
;  571          1671  2         TAPE_CHAR       : BBLOCK [$BYTEOFFSET (DIB$L_DEVDEPEND)+4],
;  572          1672  2         DESC            : VECTOR[2];
;  573          1673  2
;  574          1674  2 DESC[0] = %ALLOCATION(TAPE_CHAR);
;  575          1675  2 DESC[1] = TAPE_CHAR;
;  576        P 1676  2 $GETCHN (CHAN = .RWSV_CHAN,
;  577          1677  2          PRIBUF = DESC);
;  578          1678  2
;  579          1679  2 .TAPE_CHAR[DIB$L_DEVDEPEND]
;  580          1680  1 END;                                   ! End of routine SENSE_CHAR


                                                        .EXTRN  SYS$GETCHN

                                          0000 00000    .ENTRY  SENSE_CHAR, Save nothing                ; 1638
                                    5E    10   C2 00002    SUBL2   #16, SP                             ;
                                          0C   DD 00005    PUSHL   #12                                 ; 1674
                                04  AE    08   AE 9E 00007    MOVAB   TAPE_CHAR, DESC+4                ; 1675
                                          7E   7C 0000C    CLRQ    -(SP)                               ; 1677
                                    08    AE   9F 0000E    PUSHAB  DESC                                ;
                                          7E   D4 00011    CLRL    -(SP)                               ;
                                00000000' EF   DD 00013    PUSHL   RWSV_CHAN                           ;
```

```
                        00000000G  00            05  FB 00019          CALLS   #5, SYS$GETCHN                          ;
                                   50       10   AE  D0 00020          MOVL    TAPE_CHAR+8, R0                        ; 1679
                                                    04 00024          RET                                            ; 1680

; Routine Size:  37 bytes,    Routine Base:  CODE + 0335
```

```
;  582          1681  1 %SBTTL 'WRITE_MESSAGE - write prompt to terminal'
;  583          1682  1 ROUTINE WRITE_MESSAGE (MESSAGE,REPLY_DESC) =
;  584          1683  1
;  585          1684  1 !++
;  586          1685  1 !
;  587          1686  1 ! FUNCTIONAL DESCRIPTION:
;  588          1687  1 !
;  589          1688  1 !       This routine outputs the specified prompt message to SYS$COMMAND
;  590          1689  1 !       and waits for a response.
;  591          1690  1 !
;  592          1691  1 ! CALLING SEQUENCE:
;  593          1692  1 !       WRITE_MESSAGE (MESSAGE,REPLY_DESC)
;  594          1693  1 !
;  595          1694  1 ! INPUT PARAMETERS:
;  596          1695  1 !       MESSAGE: string descriptor of message to use
;  597          1696  1 !       REPLY_DESC: descriptor for reply buffer
;  598          1697  1 !
;  599          1698  1 ! IMPLICIT INPUTS:
;  600          1699  1 !       NONE
;  601          1700  1 !
;  602          1701  1 ! OUTPUT PARAMETERS:
;  603          1702  1 !       NONE
;  604          1703  1 !
;  605          1704  1 ! IMPLICIT OUTPUTS:
;  606          1705  1 !       NONE
;  607          1706  1 !
;  608          1707  1 ! ROUTINE VALUE:
;  609          1708  1 !       FALSE (to inhibit $PUTMSG output)
;  610          1709  1 !
;  611          1710  1 ! SIDE EFFECTS:
;  612          1711  1 !       NONE
;  613          1712  1 !
;  614          1713  1 !--
;  615          1714  1
;  616          1715  2 BEGIN
;  617          1716  2
;  618          1717  2 EXTERNAL ROUTINE
;  619          1718  2         LIB$GET_COMMAND : ADDRESSING_MODE (GENERAL);
;  620          1719  2
;  621          1720  2
;  622          1721  2 ! Use the message as the prompt to read an input line.
;  623          1722  2 !
;  624          1723  2 LIB$GET_COMMAND(.REPLY_DESC, .MESSAGE);
;  625          1724  2 FALSE
;  626          1725  1 END;                                   ! End of routine WRITE_MESSAGE


                                                        .EXTRN  LIB$GET_COMMAND

                                        0000 00000 WRITE_MESSAGE:
                                                        .WORD   Save nothing                            ; 1682
                                    04  AC  DD 00002     PUSHL   MESSAGE                                 ; 1723
                                    08  AC  DD 00005     PUSHL   REPLY_DESC                              ;
                        00000000G   00      02  FB 00008 CALLS   #2, LIB$GET_COMMAND                     ;
                                            50  D4 0000F CLRL    R0                                    ; 1725
                                                04 00011 RET                                            ;
```

; Routine Size: 18 bytes,   Routine Base: CODE + 035A

```
;  628          1726  1 %SBTTL 'MOUNT_MESSAGE - prompt for volume mount'
;  629          1727  1 GLOBAL ROUTINE MOUNT_MESSAGE (MESSAGE) : NOVALUE =
;  630          1728  1
;  631          1729  1 !++
;  632          1730  1 !
;  633          1731  1 ! FUNCTIONAL DESCRIPTION:
;  634          1732  1 !
;  635          1733  1 !       This routine issues an operator message to mount the next
;  636          1734  1 !       volume and waits for the reply.
;  637          1735  1 !
;  638          1736  1 ! CALLING SEQUENCE:
;  639          1737  1 !       MOUNT_MESSAGE (MESSAGE)
;  640          1738  1 !
;  641          1739  1 ! INPUT PARAMETERS:
;  642          1740  1 !       MESSAGE: code of message to issue
;  643          1741  1 !
;  644          1742  1 ! IMPLICIT INPUTS:
;  645          1743  1 !       NONE
;  646          1744  1 !
;  647          1745  1 ! OUTPUT PARAMETERS:
;  648          1746  1 !       NONE
;  649          1747  1 !
;  650          1748  1 ! IMPLICIT OUTPUTS:
;  651          1749  1 !       NONE
;  652          1750  1 !
;  653          1751  1 ! ROUTINE VALUE:
;  654          1752  1 !       NONE
;  655          1753  1 !
;  656          1754  1 ! SIDE EFFECTS:
;  657          1755  1 !       NONE
;  658          1756  1 !
;  659          1757  1 !--
;  660          1758  1
;  661          1759  2 BEGIN
;  662          1760  2
;  663          1761  2 LOCAL
;  664          1762  2         MSG_VECTOR      : VECTOR [6],   ! message arg vector for $PUTMSG
;  665          1763  2         REPLY_DESC      : VECTOR [2],   ! descriptor for reply
;  666          1764  2         REPLY_BUFFER    : VECTOR [4, BYTE]; ! buffer for response
;  667          1765  2
;  668          1766  2
;  669          1767  2 ! Set up the message vector and issue the message to the terminal,
;  670          1768  2 ! or to OPCOM if running under batch.
;  671          1769  2 !
;  672          1770  2
;  673          1771  2 MSG_VECTOR[0] = 5;
;  674          1772  2 MSG_VECTOR[1] = .MESSAGE;
;  675          1773  2 MSG_VECTOR[2] = 3;
;  676          1774  2 MSG_VECTOR[3] = .RWSV_VOL_NUMBER;
;  677          1775  2 MSG_VECTOR[4] = .BBLOCK[RWSV_SAVE_QUAL[QUAL_DVI_DESC], DSC$W_LENGTH];
;  678          1776  2 MSG_VECTOR[5] = .BBLOCK[RWSV_SAVE_QUAL[QUAL_DVI_DESC], DSC$A_POINTER];
;  679          1777  2 IF NOT .COM_FLAGS[COM_INTERACT]
;  680          1778  2 THEN
;  681          1779  2     OPCOM(0,
;  682          1780  2         .MSG_VECTOR[1],
;  683          1781  2         .MSG_VECTOR[3], .MSG_VECTOR[4], .MSG_VECTOR[5])
;  684          1782  2 ELSE
```

```
;  685          1783  2      WHILE TRUE
;  686          1784  2      DO
;  687          1785  3          BEGIN
;  688          1786  3          REPLY_DESC[0] = 4;
;  689          1787  3          REPLY_DESC[1] = REPLY_BUFFER;
;  690          1788  3          $PUTMSG (MSGVEC = MSG_VECTOR, ACTRTN = WRITE_MESSAGE, ACTPRM = REPLY_DESC);
;  691          1789  3          REPLY_BUFFER[0] = .REPLY_BUFFER[0] AND NOT %O'040';
;  692          1790  3          IF .REPLY_BUFFER[0] EQL 'Y'
;  693          1791  3          THEN EXIT[OOP;
;  694          1792  2          END;
;  695          1793  2
;  696          1794  1 END;                                    ! End of routine MOUNT_MESSAGE


                                        0004 00000      .ENTRY  MOUNT_MESSAGE, Save R2                          ; 1727
                    52 00000000'    EF   9E 00002       MOVAB   RWSV_VOL_NUMBER, R2                             ;
                    5E              24   C2 00009       SUBL2   #36, SP                                         ;
                0C  AE              05   D0 0000C       MOVL    #5, MSG_VECTOR                                  ; 1771
                10  AE          04  AC   D0 00010       MOVL    MESSAGE, MSG_VECTOR+4                           ; 1772
                14  AE              03   D0 00015       MOVL    #3, MSG_VECTOR+8                                ; 1773
                18  AE              62   3C 00019       MOVZWL  RWSV_VOL_NUMBER, MSG_VECTOR+12                  ; 1774
                    50          08  A2   D0 0001D       MOVL    RWSV_SAVE_QUAL, R0                              ; 1775
                    50              18   C0 00021       ADDL2   #24, R0                                         ;
                1C  AE              60   3C 00024       MOVZWL  (R0), MSG_VECTOR+16                             ;
                20  AE          04  A0   D0 00028       MOVL    4(R0), MSG_VECTOR+20                            ; 1776
                    14      00DB    C2   E8 0002D       BLBS    COM_FLAGS+1, 1$                                 ; 1777
                            20  AE  DD   00032         PUSHL   MSG_VECTOR+20                                   ; 1781
                            20  AE  DD   00035         PUSHL   MSG_VECTOR+16                                   ;
                            20  AE  DD   00038         PUSHL   MSG_VECTOR+12                                   ;
                            1C  AE  DD   0003B         PUSHL   MSG_VECTOR+4                                    ; 1780
                                7E  D4   0003E         CLRL    -(SP)                                           ; 1779
                        FC4F    CF      05   FB 00040       CALLS   #5, OPCOM                                   ;
                                        04   00045       RET                                                     ;
                            04  AE      04   D0 00046 1$:   MOVL    #4, REPLY_DESC                              ; 1786
                            08  AE      6E   9E 0004A       MOVAB   REPLY_BUFFER, REPLY_DESC+4                  ; 1787
                                    04  AE   9F 0004E       PUSHAB  REPLY_DESC                                  ; 1788
                                        7E   D4 00051       CLRL    -(SP)                                       ;
                                    98  AF   9F 00053       PUSHAB  WRITE_MESSAGE                               ;
                                    18  AE   9F 00056       PUSHAB  MSG_VECTOR                                  ;
                        00000000G   00      04   FB 00059       CALLS   #4, SYS$PUTMSG                          ;
                                    6E      20   8A 00060       BICB2   #32, REPLY_BUFFER                       ; 1789
                                59  8F      6E   91 00063       CMPB    REPLY_BUFFER, #89                       ; 1790
                                        DD   12   00067       BNEQ    1$                                      ;
                                        04   00069       RET                                                     ; 1794

; Routine Size:  106 bytes,    Routine Base:  CODE + 036C
```

```
;  698          1795  1 %SBTTL 'READY_TAPE - make tape ready'
;  699          1796  1 GLOBAL ROUTINE READY_TAPE (WRITE) =
;  700          1797  1
;  701          1798  1 !++
;  702          1799  1 !
;  703          1800  1 ! FUNCTIONAL DESCRIPTION:
;  704          1801  1 !
;  705          1802  1 !       This routine gets the tape ready as specified and returns the
;  706          1803  1 !       tape device characteristics.
;  707          1804  1 !
;  708          1805  1 ! CALLING SEQUENCE:
;  709          1806  1 !       READY_TAPE (WRITE)
;  710          1807  1 !
;  711          1808  1 ! INPUT PARAMETERS:
;  712          1809  1 !       WRITE: FALSE if tape is to be read
;  713          1810  1 !              TRUE if tape is to be written
;  714          1811  1 !
;  715          1812  1 ! IMPLICIT INPUTS:
;  716          1813  1 !       NONE
;  717          1814  1 !
;  718          1815  1 ! OUTPUT PARAMETERS:
;  719          1816  1 !       NONE
;  720          1817  1 !
;  721          1818  1 ! IMPLICIT OUTPUTS:
;  722          1819  1 !       NONE
;  723          1820  1 !
;  724          1821  1 ! ROUTINE VALUE:
;  725          1822  1 !       tape device characteristics longword
;  726          1823  1 !
;  727          1824  1 ! SIDE EFFECTS:
;  728          1825  1 !       NONE
;  729          1826  1 !
;  730          1827  1 !--
;  731          1828  1
;  732          1829  2 BEGIN
;  733          1830  2
;  734          1831  2 LOCAL
;  735          1832  2         STATUS,                         ! system service status
;  736          1833  2         MSG_VECTOR      : VECTOR [6],   ! message arg vector for $PUTMSG
;  737          1834  2         IO_STATUS       : VECTOR [4,WORD]; ! I/O status block
;  738          1835  2
;  739          1836  2 EXTERNAL ROUTINE
;  740          1837  2         FILE_ERROR;                     ! signal file related error
;  741          1838  2
;  742          1839  2
;  743          1840  2 ! Loop, checking for tape on line and write enabled if necessary,
;  744          1841  2 ! prompting to the user until satisfied.
;  745          1842  2 !
;  746          1843  2
;  747          1844  2 WHILE TRUE
;  748          1845  2 DO
;  749          1846  3     BEGIN
;  750        P 1847  3     STATUS = $QIOW (CHAN = .RWSV_CHAN,
;  751        P 1848  3                     FUNC = IO$_SENSEMODE,
;  752        P 1849  3                     IOSB = IO_STATUS
;  753          1850  3                     );
;  754          1851  3     IF .STATUS THEN STATUS = .IO_STATUS[0];
```

```
;  755          1852  3
;  756          1853  3     IF .STATUS EQL SS$_ENDOFTAPE
;  757          1854  3     THEN STATUS = TRUE;
;  758          1855  3
;  759          1856  3     IF .STATUS EQL SS$_MEDOFL
;  760          1857  3     OR .STATUS EQL SS$_VOLINV   ! **** Temp until TMDRIVER is fixed
;  761          1858  3     THEN
;  762          1859  4         BEGIN
;  763          1860  4         IF .WRITE
;  764          1861  4         THEN MOUNT_MESSAGE (BACKUP$_READYWRITE)
;  765          1862  4         ELSE MOUNT_MESSAGE (BACKUP$_READYREAD);
;  766          1863  4         END
;  767          1864  4
;  768          1865  3     ELSE IF NOT .STATUS
;  769          1866  3     THEN FILE_ERROR (BACKUP$_LABELERR, .RWSV_SAVE_FAB, .STATUS)
;  770          1867  3
;  771          1868  3     ELSE IF .WRITE AND .BBLOCK [IO_STATUS[2], MT$V_HWL]
;  772          1869  3     THEN MOUNT_MESSAGE (BACKUP$_WRITENABLE)
;  773          1870  3
;  774          1871  3     ELSE EXITLOOP;
;  775          1872  2     END;
;  776          1873  2
;  777          1874  2 SENSE_CHAR ()
;  778          1875  1 END;                                   ! End of routine READY_TAPE


                                                        .EXTRN  FILE_ERROR

                                          0004 00000    .ENTRY  READY_TAPE, Save R2                             ; 1796
                    5E                        20  C2 00002    SUBL2   #32, SP                                 ;
                                              7E  7C 00005 1$:  CLRQ    -(SP)                                   ; 1850
                                              7E  7C 00007      CLRQ    -(SP)                                   ;
                                              7E  7C 00009      CLRQ    -(SP)                                   ;
                                              7E  7C 0000B      CLRQ    -(SP)                                   ;
                                  20          AE  9F 0000D      PUSHAB  IO_STATUS                               ;
                                              27  DD 00010      PUSHL   #39                                     ;
                        00000000'             EF  DD 00012      PUSHL   RWSV_CHAN                               ;
                                              7E  D4 00018      CLRL    -(SP)                                   ;
000000000G              00                    0C  FB 0001A      CALLS   #12, SYS$QIOW                           ;
                        52                    50  D0 00021      MOVL    R0, STATUS                              ;
                        03                    52  E9 00024      BLBC    STATUS, 2$                              ; 1851
                        52                    6E  3C 00027      MOVZWL  IO_STATUS, STATUS                       ;
00000878                8F                    52  D1 0002A 2$:  CMPL    STATUS, #2168                           ; 1853
                                              03  12 00031      BNEQ    3$                                      ;
                        52                    01  D0 00033      MOVL    #1, STATUS                              ; 1854
000001A4                8F                    52  D1 00036 3$:  CMPL    STATUS, #420                            ; 1856
                                              09  13 0003D      BEQL    4$                                      ;
00000254                8F                    52  D1 0003F      CMPL    STATUS, #596                            ; 1857
                                              14  12 00046      BNEQ    6$                                      ;
                        08          04        AC  E9 00048 4$:  BLBC    WRITE, 5$                               ; 1860
                              00000000G       8F  DD 0004C      PUSHL   #BACKUP$_READYWRITE                     ; 1861
                                              31  11 00052      BRB     9$                                      ;
                              00000000G       8F  DD 00054 5$:  PUSHL   #BACKUP$_READYREAD                      ; 1862
                                              29  11 0005A      BRB     9$                                      ;
                        17                    52  E8 0005C 6$:  BLBS    STATUS, 8$                              ; 1865
                                              52  DD 0005F      PUSHL   STATUS                                  ; 1866
```

```
                                00000000'  EF  DD 00061          PUSHL   RWSV_SAVE_FAB                    ;
                                00000000G  8F  DD 00067          PUSHL   #BACKUP$_LABELERR               ;
                     00000000G  00          03  FB 0006D          CALLS   #3, FILE_ERROR                  ;
                                            8F  11 00074 7$:      BRB     1$                              ;
                                12      04  AC  E9 00076 8$:      BLBC    WRITE, 10$                      ; 1868
                OD      06      AE          03  E1 0007A          BBC     #3, IO_STATUS+6, 10$            ;
                                00000000G  8F  DD 0007F          PUSHL   #BACKUP$_WRITENABLE             ; 1869
                        FF0C    CF          01  FB 00085 9$:      CALLS   #1, MOUNT_MESSAGE               ;
                                            E8  11 0008A          BRB     7$                              ;
                        FECE    CF          00  FB 0008C 10$:     CALLS   #0, SENSE_CHAR                  ; 1874
                                            04  00091          RET                                     ; 1875

; Routine Size:  146 bytes,    Routine Base:  CODE + 03D6
```

```
;  780          1876  1 %SBTTL 'SET_CHAR - set tape characteristics'
;  781          1877  1 GLOBAL ROUTINE SET_CHAR (CHAR) : NOVALUE =
;  782          1878  1
;  783          1879  1 !++
;  784          1880  1 !
;  785          1881  1 ! FUNCTIONAL DESCRIPTION:
;  786          1882  1 !
;  787          1883  1 !       This routine sets the tape characteristics.
;  788          1884  1 !
;  789          1885  1 ! CALLING SEQUENCE:
;  790          1886  1 !       SET_CHAR (CHAR)
;  791          1887  1 !
;  792          1888  1 ! INPUT PARAMETERS:
;  793          1889  1 !       CHAR: tape characteristics word to set
;  794          1890  1 !
;  795          1891  1 ! IMPLICIT INPUTS:
;  796          1892  1 !       NONE
;  797          1893  1 !
;  798          1894  1 ! OUTPUT PARAMETERS:
;  799          1895  1 !       NONE
;  800          1896  1 !
;  801          1897  1 ! IMPLICIT OUTPUTS:
;  802          1898  1 !       NONE
;  803          1899  1 !
;  804          1900  1 ! ROUTINE VALUE:
;  805          1901  1 !       tape device characteristics longword
;  806          1902  1 !       NONE
;  807          1903  1 ! SIDE EFFECTS:
;  808          1904  1 !       NONE
;  809          1905  1 !
;  810          1906  1 !--
;  811          1907  1
;  812          1908  2 BEGIN
;  813          1909  2
;  814          1910  2 LOCAL
;  815          1911  2         TAPE_CHAR       : BBLOCK [$BYTEOFFSET (DIB$L_DEVDEPEND)+4],
;  816          1912  2         DESC            : VECTOR[2],
;  817          1913  2         STATUS,                         ! system service status
;  818          1914  2         IO_STATUS       : VECTOR [4,WORD]; ! I/O status block
;  819          1915  2
;  820          1916  2 EXTERNAL ROUTINE
;  821          1917  2         FILE_ERROR;                     ! signal file related error
;  822          1918  2
;  823          1919  2 WHILE TRUE
;  824          1920  2 DO
;  825          1921  3     BEGIN
;  826          1922  3     DESC[0] = %ALLOCATION(TAPE_CHAR);
;  827          1923  3     DESC[1] = TAPE_CHAR;
;  828        P 1924  3     $GETCHN (CHAN = .RWSV_CHAN,
;  829          1925  3              PRIBUF = DESC);
;  830          1926  3
;  831          1927  3     TAPE_CHAR[DIB$L_DEVDEPEND] = .CHAR;
;  832        P 1928  3     STATUS = $QIOW (CHAN = .RWSV_CHAN,
;  833        P 1929  3                     FUNC = IO$_SETMODE,
;  834        P 1930  3                     IOSB = IO_STATUS,
;  835        P 1931  3                     P1   = TAPE_CHAR[DIB$B_DEVCLASS]
;  836          1932  3                     );
```

```
;  837          1933  3     IF .STATUS THEN STATUS = .IO_STATUS[0];
;  838          1934  3     IF .STATUS EQL SS$_ENDOFTAPE THEN STATUS = TRUE;
;  839          1935  3     IF .STATUS THEN EXITLOOP;
;  840          1936  3     FILE_ERROR (BACKUP$_POSITERR, .RWSV_SAVE_FAB, .STATUS);
;  841          1937  2     END;
;  842          1938  2
;  843          1939  1 END;                                    ! End of routine SET_CHAR


                                    000C 00000          .ENTRY  SET_CHAR, Save R2,R3                    ; 1877
                         53 00000000'   EF  9E 00002    MOVAB   RWSV_CHAN, R3                           ;
                         5E             1C  C2 00009    SUBL2   #28, SP                                 ;
                      08 AE             0C  D0 0000C 1$:  MOVL  #12, DESC                               ; 1922
                      0C AE       10    AE  9E 00010    MOVAB   TAPE_CHAR, DESC+4                       ; 1923
                                  7E    7C  00015       CLRQ    -(SP)                                   ; 1925
                                  10 AE 9F  00017       PUSHAB  DESC                                    ;
                                  7E    D4  0001A       CLRL    -(SP)                                   ;
                                  63    DD  0001C       PUSHL   RWSV_CHAN                               ;
                00000000G         00    05  FB  0001E   CALLS   #5, SYS$GETCHN                         ;
                      18 AE       04    AC  D0  00025   MOVL    CHAR, TAPE_CHAR+8                       ; 1927
                                  7E    7C  0002A       CLRQ    -(SP)                                   ; 1932
                                  7E    7C  0002C       CLRQ    -(SP)                                   ;
                                  7E    D4  0002E       CLRL    -(SP)                                   ;
                                  28 AE 9F  00030       PUSHAB  TAPE_CHAR+4                             ;
                                  7E    7C  00033       CLRQ    -(SP)                                   ;
                                  20 AE 9F  00035       PUSHAB  IO_STATUS                               ;
                                  23    DD  00038       PUSHL   #35                                     ;
                                  63    DD  0003A       PUSHL   RWSV_CHAN                               ;
                                  7E    D4  0003C       CLRL    -(SP)                                   ;
                00000000G         00    0C  FB  0003E   CALLS   #12, SYS$QIOW                           ;
                                  52    50  D0  00045   MOVL    R0, STATUS                              ;
                                  03    52  E9  00048   BLBC    STATUS, 2$                              ; 1933
                                  52    6E  3C  0004B   MOVZWL  IO_STATUS, STATUS                      ;
                00000878          8F    52  D1  0004E 2$: CMPL  STATUS, #2168                           ; 1934
                                        03  12  00055   BNEQ    3$                                      ;
                                  52    01  D0  00057   MOVL    #1, STATUS                              ;
                                  14    52  E8  0005A 3$: BLBS  STATUS, 4$                              ; 1935
                                        52  DD  0005D   PUSHL   STATUS                                  ; 1936
                                     FC 53  A3  DD  0005F   PUSHL   RWSV_SAVE_FAB                       ;
                                00000000G   8F  DD  00062   PUSHL   #BACKUP$_POSITERR                   ;
                00000000G         00        03  FB  00068   CALLS   #3, FILE_ERROR                     ;
                                            9B  11  0006F   BRB     1$                                  ; 1919
                                                04  00071 4$: RET                                       ; 1939

; Routine Size:  114 bytes,    Routine Base:  CODE + 0468
```

```
;  845          1940  1 %SBTTL 'REWIND - rewind tape'
;  846          1941  1 GLOBAL ROUTINE REWIND : NOVALUE =
;  847          1942  1
;  848          1943  1 !++
;  849          1944  1 !
;  850          1945  1 ! FUNCTIONAL DESCRIPTION:
;  851          1946  1 !
;  852          1947  1 !       This routine rewinds the save set tape.
;  853          1948  1 !
;  854          1949  1 ! CALLING SEQUENCE:
;  855          1950  1 !       REWIND ()
;  856          1951  1 !
;  857          1952  1 ! INPUT PARAMETERS:
;  858          1953  1 !       NONE
;  859          1954  1 !
;  860          1955  1 ! IMPLICIT INPUTS:
;  861          1956  1 !       NONE
;  862          1957  1 !
;  863          1958  1 ! OUTPUT PARAMETERS:
;  864          1959  1 !       NONE
;  865          1960  1 !
;  866          1961  1 ! IMPLICIT OUTPUTS:
;  867          1962  1 !       NONE
;  868          1963  1 !
;  869          1964  1 ! ROUTINE VALUE:
;  870          1965  1 !       NONE
;  871          1966  1 !
;  872          1967  1 ! SIDE EFFECTS:
;  873          1968  1 !       NONE
;  874          1969  1 !
;  875          1970  1 !--
;  876          1971  1
;  877          1972  2 BEGIN
;  878          1973  2
;  879          1974  2 LOCAL
;  880          1975  2         STATUS,                         ! general status value
;  881          1976  2         IO_STATUS       : VECTOR [4, WORD]; ! I/O status block
;  882          1977  2
;  883          1978  2 EXTERNAL ROUTINE
;  884          1979  2         FILE_ERROR;                     ! signal file related error
;  885          1980  2
;  886          1981  2 WHILE TRUE
;  887          1982  2 DO
;  888          1983  3     BEGIN
;  889        P 1984  3     STATUS = $QIOW (CHAN = .RWSV_CHAN,
;  890        P 1985  3                     FUNC = IO$_REWIND,
;  891        P 1986  3                     IOSB = IO_STATUS
;  892          1987  3                     );
;  893          1988  3     IF .STATUS THEN STATUS = .IO_STATUS[0];
;  894          1989  3     IF .STATUS EQL SS$_ENDOFTAPE THEN STATUS = TRUE;
;  895          1990  3     IF .STATUS THEN EXITLOOP;
;  896          1991  3     FILE_ERROR (BACKUP$_POSITERR, .RWSV_SAVE_FAB, .STATUS);
;  897          1992  2     END;
;  898          1993  2
;  899          1994  1 END;                                    ! End of routine REWIND
```

```
                                        0004 00000          .ENTRY  REWIND, Save r2                 ; 1941
                                5E      08   C2 00002          SUBL2   #8, SP
                                        7E   7C 00005 1$:      CLRQ    -(SP)                       ; 1987
                                        7E   7C 00007          CLRQ    -(SP)
                                        7E   7C 00009          CLRQ    -(SP)
                                        7E   7C 0000B          CLRQ    -(SP)
                                   20   AE   9F 0000D          PUSHAB  IO_STATUS
                                        24   DD 00010          PUSHL   #36
                            00000000'   EF   DD 00012          PUSHL   RWSV_CHAN
                                        7E   D4 00018          CLRL    -(SP)
                00000000G   00          0C   FB 0001A          CALLS   #12, SYS$QIOW
                            52          50   D0 00021          MOVL    R0, STATUS
                            03          52   E9 00024          BLBC    STATUS, 2$                  ; 1988
                            52          6E   3C 00027          MOVZWL  IO_STATUS, STATUS
                00000878    8F          52   D1 0002A 2$:      CMPL    STATUS, #1878               ; 1989
                                        03   12 00031          BNEQ    3$
                            52          01   D0 00033          MOVL    #1, STATUS
                            17          52   E8 00036 3$:      BLBS    STATUS, 4$                  ; 1990
                                        52   DD 00039          PUSHL   STATUS                      ; 1991
                            00000000'   EF   DD 0003B          PUSHL   RWSV_SAVE_FAB
                            00000000G   8F   DD 00041          PUSHL   #BACKUP$_POSITERR
                00000000G   00          03   FB 00047          CALLS   #3, FILE_ERROR
                                        B5   11 0004E          BRB     1$                          ; 1981
                                             04 00050 4$:      RET                                 ; 1994

; Routine Size:  81 bytes,    Routine Base:  CODE + 04DA
```

```
;  901          1995  1 %SBTTL 'UNLOAD - unload tape'
;  902          1996  1 GLOBAL ROUTINE UNLOAD : NOVALUE =
;  903          1997  1
;  904          1998  1 !++
;  905          1999  1 !
;  906          2000  1 ! FUNCTIONAL DESCRIPTION:
;  907          2001  1 !
;  908          2002  1 !       This routine unloads the save set tape.
;  909          2003  1 !
;  910          2004  1 ! CALLING SEQUENCE:
;  911          2005  1 !       UNLOAD ()
;  912          2006  1 !
;  913          2007  1 ! INPUT PARAMETERS:
;  914          2008  1 !       NONE
;  915          2009  1 !
;  916          2010  1 ! IMPLICIT INPUTS:
;  917          2011  1 !       NONE
;  918          2012  1 !
;  919          2013  1 ! OUTPUT PARAMETERS:
;  920          2014  1 !       NONE
;  921          2015  1 !
;  922          2016  1 ! IMPLICIT OUTPUTS:
;  923          2017  1 !       NONE
;  924          2018  1 !
;  925          2019  1 ! ROUTINE VALUE:
;  926          2020  1 !       NONE
;  927          2021  1 !
;  928          2022  1 ! SIDE EFFECTS:
;  929          2023  1 !       NONE
;  930          2024  1 !
;  931          2025  1 !--
;  932          2026  1
;  933          2027  2 BEGIN
;  934          2028  2
;  935          2029  2 LOCAL
;  936          2030  2         STATUS,                                 ! general status value
;  937          2031  2         IO_STATUS       : VECTOR [4, WORD]; ! I/O status block
;  938          2032  2
;  939          2033  2 EXTERNAL ROUTINE
;  940          2034  2         FILE_ERROR;                             ! signal file related error
;  941          2035  2
;  942          2036  2 WHILE TRUE
;  943          2037  2 DO
;  944          2038  3     BEGIN
;  945        P 2039  3     STATUS = $QIOW (CHAN = .RWSV_CHAN,
;  946        P 2040  3                     FUNC = IO$_UNLOAD,
;  947        P 2041  3                     IOSB = IO_STATUS
;  948          2042  3                     );
;  949          2043  3     IF .STATUS THEN STATUS = .IO_STATUS[0];
;  950          2044  3     IF NOT .STATUS
;  951          2045  3     THEN
;  952          2046  4         BEGIN
;  953          2047  4         IF .STATUS EQL SS$_NOPRIV
;  954          2048  4         THEN
;  955          2049  5             BEGIN
;  956        P 2050  5             STATUS = $QIOW (CHAN = .RWSV_CHAN,
;  957        P 2051  5                             FUNC = IO$_REWINDOFF,
```

```
;  958        P 2052  5                                IOSB = IO_STATUS
;  959          2053  5                                );
;  960          2054  5                    IF .STATUS THEN STATUS = .IO_STATUS[0];
;  961          2055  4                    END;
;  962          2056  3                END;
;  963          2057  3            IF .STATUS EQL SS$_ENDOFTAPE THEN STATUS = TRUE;
;  964          2058  3            IF .STATUS THEN EXITLOOP;
;  965          2059  3            FILE_ERROR (BACKUP$_POSITERR, .RWSV_SAVE_FAB, .STATUS);
;  966          2060  2            END;
;  967          2061  2
;  968          2062  1 END;                                        ! End of routine UNLOAD


                                                001C 00000          .ENTRY  UNLOAD, Save R2,R3,R4                   ; 1996
                                54 00000000G    00   9E 00002      MOVAB   SYS$QIOW, R4                            ;
                                53 00000000'    EF   9E 00009      MOVAB   RWSV_CHAN, R3                           ;
                                5E              08   C2 00010      SUBL2   #8, SP                                  ;
                                                7E   7C 00013 1$:   CLRQ    -(SP)                                   ; 2042
                                                7E   7C 00015      CLRQ    -(SP)                                   ;
                                                7E   7C 00017      CLRQ    -(SP)                                   ;
                                                7E   7C 00019      CLRQ    -(SP)                                   ;
                                        20      AE   9F 0001B      PUSHAB  IO_STATUS                               ;
                                                01   DD 0001E      PUSHL   #1                                      ;
                                                63   DD 00020      PUSHL   RWSV_CHAN                               ;
                                                7E   D4 00022      CLRL    -(SP)                                   ;
                                64              0C   FB 00024      CALLS   #12, SYS$QIOW                           ;
                                52              50   D0 00027      MOVL    R0, STATUS                              ;
                                06              52   E9 0002A      BLBC    STATUS, 2$                              ; 2043
                                52              6E   3C 0002D      MOVZWL  IO_STATUS, STATUS                       ;
                                22              52   E8 00030      BLBS    STATUS, 3$                              ; 2044
                                24              52   D1 00033 2$:   CMPL    STATUS, #36                             ; 2047
                                                1D   12 00036      BNEQ    3$                                      ;
                                                7E   7C 00038      CLRQ    -(SP)                                   ; 2053
                                                7E   7C 0003A      CLRQ    -(SP)                                   ;
                                                7E   7C 0003C      CLRQ    -(SP)                                   ;
                                                7E   7C 0003E      CLRQ    -(SP)                                   ;
                                        20      AE   9F 00040      PUSHAB  IO_STATUS                               ;
                                                22   DD 00043      PUSHL   #34                                     ;
                                                63   DD 00045      PUSHL   RWSV_CHAN                               ;
                                                7E   D4 00047      CLRL    -(SP)                                   ;
                                64              0C   FB 00049      CALLS   #12, SYS$QIOW                           ;
                                52              50   D0 0004C      MOVL    R0, STATUS                              ;
                                03              52   E9 0004F      BLBC    STATUS, 3$                              ; 2054
                                52              6E   3C 00052      MOVZWL  IO_STATUS, STATUS                       ;
00000878                        8F              52   D1 00055 3$:   CMPL    STATUS, #2168                           ; 2057
                                                03   12 0005C      BNEQ    4$                                      ;
                                52              01   D0 0005E      MOVL    #1, STATUS                              ;
                                14              52   E8 00061 4$:   BLBS    STATUS, 5$                              ; 2058
                                                52   DD 00064      PUSHL   STATUS                                  ; 2059
                                        FC      A3   DD 00066      PUSHL   RWSV_SAVE_FAB                           ;
                            00000000G           8F   DD 00069      PUSHL   #BACKUP$_POSITERR                       ;
00000000G                   00                  03   FB 0006F      CALLS   #3, FILE_ERROR                          ;
                                                9B   11 00076      BRB     1$                                      ; 2036
                                                     04 00078 5$:   RET                                             ; 2062
```

; Routine Size: 121 bytes,   Routine Base: CODE + 052B

```
;  970          2063  1 %SBTTL 'SKIP_TM - skip tape marks'
;  971          2064  1 GLOBAL ROUTINE SKIP_TM (COUNT) =
;  972          2065  1
;  973          2066  1 !++
;  974          2067  1 !
;  975          2068  1 ! FUNCTIONAL DESCRIPTION:
;  976          2069  1 !
;  977          2070  1 !       This routine skips the specified number of tape marks
;  978          2071  1 !       forward or backward on the tape.
;  979          2072  1 !
;  980          2073  1 ! CALLING SEQUENCE:
;  981          2074  1 !       SKIP_TM (COUNT)
;  982          2075  1 !
;  983          2076  1 ! INPUT PARAMETERS:
;  984          2077  1 !       COUNT: number of tape marks to skip, + for forward, - for reverse
;  985          2078  1 !
;  986          2079  1 ! IMPLICIT INPUTS:
;  987          2080  1 !       NONE
;  988          2081  1 !
;  989          2082  1 ! OUTPUT PARAMETERS:
;  990          2083  1 !       NONE
;  991          2084  1 !
;  992          2085  1 ! IMPLICIT OUTPUTS:
;  993          2086  1 !       NONE
;  994          2087  1 !
;  995          2088  1 ! ROUTINE VALUE:
;  996          2089  1 !       TRUE if success
;  997          2090  1 !       SS$_ENDOFVOLUME if 2 successive tape marks encountered
;  998          2091  1 !
;  999          2092  1 ! SIDE EFFECTS:
; 1000          2093  1 !       NONE
; 1001          2094  1 !
; 1002          2095  1 !--
; 1003          2096  1
; 1004          2097  2 BEGIN
; 1005          2098  2
; 1006          2099  2 LOCAL
; 1007          2100  2         STATUS,                                 ! general status value
; 1008          2101  2         IO_STATUS       : VECTOR [4, WORD]; ! I/O status block
; 1009          2102  2
; 1010          2103  2 EXTERNAL ROUTINE
; 1011          2104  2         FILE_ERROR;                             ! signal file related error
; 1012          2105  2
; 1013        P 2106  2 STATUS = $QIOW (CHAN = .RWSV_CHAN,
; 1014        P 2107  2                 FUNC = IO$_SKIPFILE,
; 1015        P 2108  2                 IOSB = IO_STATUS,
; 1016        P 2109  2                 P1   = .COUNT
; 1017          2110  2                 );
; 1018          2111  2 IF .STATUS THEN STATUS = .IO_STATUS[0];
; 1019          2112  2 IF .STATUS EQL SS$_ENDOFTAPE
; 1020          2113  2 THEN STATUS = TRUE;
; 1021          2114  2 IF NOT .STATUS
; 1022          2115  2 AND .STATUS NEQ SS$_ENDOFVOLUME
; 1023          2116  2 THEN FILE_ERROR (BACKUP$_POSITERR, .RWSV_SAVE_FAB, .STATUS);
; 1024          2117  2
; 1025          2118  2 .STATUS
; 1026          2119  1 END;                                            ! End of routine SKIP_TM
```

```
                                  0004 00000         .ENTRY  SKIP_TM, Save R2                        ;2064
                          5E        08 C2 00002      SUBL2   #8, SP                                  ;
                                    7E 7C 00005      CLRQ    -(SP)                                   ;2110
                                    7E 7C 00007      CLRQ    -(SP)                                   ;
                                    7E D4 00009      CLRL    -(SP)                                   ;
                              04    AC DD 0000B      PUSHL   COUNT                                   ;
                                    7E 7C 0000E      CLRQ    -(SP)                                   ;
                              20    AE 9F 00010      PUSHAB  IO_STATUS                               ;
                                    25 DD 00013      PUSHL   #37                                     ;
                        00000000'   EF DD 00015      PUSHL   RWSV_CHAN                               ;
                                    7E D4 0001B      CLRL    -(SP)                                   ;
                00000000G   00      0C FB 0001D      CALLS   #12, SYS$QIOW                           ;
                            52      50 D0 00024      MOVL    R0, STATUS                              ;
                            03      52 E9 00027      BLBC    STATUS, 1$                              ;2111
                            52      6E 3C 0002A      MOVZWL  IO_STATUS, STATUS                       ;
                00000878    8F      52 D1 0002D 1$:  CMPL    STATUS, #2168                           ;2112
                                    03 12 00034      BNEQ    2$                                      ;
                            52      01 D0 00036      MOVL    #1, STATUS                              ;2113
                            1E      52 E8 00039 2$:  BLBS    STATUS, 3$                              ;2114
                000009A0    8F      52 D1 0003C      CMPL    STATUS, #2464                           ;2115
                                    15 13 00043      BEQL    3$                                      ;
                                    52 DD 00045      PUSHL   STATUS                                  ;2116
                        00000000'   EF DD 00047      PUSHL   RWSV_SAVE_FAB                           ;
                        00000000G   8F DD 0004D      PUSHL   #BACKUP$_POSITERR                       ;
                00000000G   00      03 FB 00053      CALLS   #3, FILE_ERROR                          ;
                            50      52 D0 0005A 3$:  MOVL    STATUS, R0                              ;2119
                                    04 0005D         RET                                             ;
                                                                                                     ;

; Routine Size:  94 bytes,    Routine Base:  CODE + 05A4
```

```
;1028          2120  1 %SBTTL 'SKIP_RECORD - skip tape records'
;1029          2121  1 GLOBAL ROUTINE SKIP_RECORD (COUNT) =
;1030          2122  1
;1031          2123  1 !++
;1032          2124  1 !
;1033          2125  1 ! FUNCTIONAL DESCRIPTION:
;1034          2126  1 !
;1035          2127  1 !       This routine skips the specified number of records
;1036          2128  1 !       forward or backward on the tape.
;1037          2129  1 !
;1038          2130  1 ! CALLING SEQUENCE:
;1039          2131  1 !       SKIP_RECORD (COUNT)
;1040          2132  1 !
;1041          2133  1 ! INPUT PARAMETERS:
;1042          2134  1 !       COUNT: number of records to skip, + for forward, - for reverse
;1043          2135  1 !
;1044          2136  1 ! IMPLICIT INPUTS:
;1045          2137  1 !       NONE
;1046          2138  1 !
;1047          2139  1 ! OUTPUT PARAMETERS:
;1048          2140  1 !       NONE
;1049          2141  1 !
;1050          2142  1 ! IMPLICIT OUTPUTS:
;1051          2143  1 !       NONE
;1052          2144  1 !
;1053          2145  1 ! ROUTINE VALUE:
;1054          2146  1 !       TRUE if success
;1055          2147  1 !       SS$_ENDOFFILE if a tape mark is encountered
;1056          2148  1 !
;1057          2149  1 ! SIDE EFFECTS:
;1058          2150  1 !       NONE
;1059          2151  1 !
;1060          2152  1 !--
;1061          2153  1
;1062          2154  2 BEGIN
;1063          2155  2
;1064          2156  2 LOCAL
;1065          2157  2         DUMMY           : BBLOCK [16],  ! dummy I/O buffer for read
;1066          2158  2         STATUS,                         ! general status value
;1067          2159  2         IO_STATUS       : VECTOR [4, WORD]; ! I/O status block
;1068          2160  2
;1069          2161  2 EXTERNAL ROUTINE
;1070          2162  2         FILE_ERROR;                     ! signal file related error
;1071          2163  2
;1072          2164  2 ! To avoid some crocks in the various magtape drivers' EOV handling,
;1073          2165  2 ! if the record count is 1, read it rather than skipping.
;1074          2166  2 !
;1075          2167  2
;1076          2168  2 IF .COUNT EQL 1
;1077          2169  2 THEN
;1078        P 2170  2     STATUS = $QIOW (CHAN = .RWSV_CHAN,
;1079        P 2171  2                     FUNC = IO$_READLBLK,
;1080        P 2172  2                     IOSB = IO_STATUS,
;1081        P 2173  2                     P1   = DUMMY,
;1082        P 2174  2                     P2   = 16
;1083          2175  3                     )
;1084          2176  2 ELSE
```

```
;  1085        P 2177  2     STATUS = $QIOW (CHAN = .RWSV_CHAN,
;  1086        P 2178  2                     FUNC = IO$_SKIPRECORD,
;  1087        P 2179  2                     IOSB = IO_STATUS,
;  1088        P 2180  2                     P1   = .COUNT
;  1089          2181  2                     );
;  1090          2182  2 IF .STATUS THEN STATUS = .IO_STATUS[0];
;  1091          2183  2 IF .STATUS EQL SS$_ENDOFTAPE
;  1092          2184  2 OR .STATUS EQL SS$_DATAOVERUN
;  1093          2185  2 THEN STATUS = TRUE;
;  1094          2186  2 IF NOT .STATUS
;  1095          2187  2 AND .STATUS NEQ SS$_ENDOFFILE
;  1096          2188  2 THEN FILE_ERROR (BACKUP$_POSITERR, .RWSV_SAVE_FAB, .STATUS);
;  1097          2189  2
;  1098          2190  2 .STATUS
;  1099          2191  1 END;                                    ! End of routine SKIP_RECORD


                                  0004 00000         .ENTRY  SKIP_RECORD, Save R2                    ; 2121
                              5E      18   C2 00002         SUBL2   #24, SP                         ;
                              01      04   AC   D1 00005         CMPL    COUNT, #1                  ; 2168
                                           12   12 00009         BNEQ    1$                         ;
                                           7E   7C 0000B         CLRQ    -(SP)                      ; 2175
                                           7E   7C 0000D         CLRQ    -(SP)                      ;
                                           10   DD 0000F         PUSHL   #16                        ;
                                      1C   AE   9F 00011         PUSHAB  DUMMY                      ;
                                           7E   7C 00014         CLRQ    -(SP)                      ;
                                      20   AE   9F 00016         PUSHAB  IO_STATUS                  ;
                                           21   DD 00019         PUSHL   #33                        ;
                                           10   11 0001B         BRB     2$                         ;
                                           7E   7C 0001D 1$:     CLRQ    -(SP)                      ; 2181
                                           7E   7C 0001F         CLRQ    -(SP)                      ;
                                           7E   D4 00021         CLRL    -(SP)                      ;
                                      04   AC   DD 00023         PUSHL   COUNT                      ;
                                           7E   7C 00026         CLRQ    -(SP)                      ;
                                      20   AE   9F 00028         PUSHAB  IO_STATUS                  ;
                                           26   DD 0002B         PUSHL   #38                        ;
                                00000000'  EF   DD 0002D 2$:     PUSHL   RWSV_CHAN                  ;
                                           7E   D4 00033         CLRL    -(SP)                      ;
                00000000G     00           0C   FB 00035         CALLS   #12, SYS$QIOW              ;
                              52           50   D0 0003C         MOVL    R0, STATUS                 ;
                              03           52   E9 0003F         BLBC    STATUS, 3$                 ; 2182
                              52           6E   3C 00042         MOVZWL  IO_STATUS, STATUS          ;
                00000878      8F           52   D1 00045 3$:     CMPL    STATUS, #2168              ; 2183
                                           09   13 0004C         BEQL    4$                         ;
                00000838      8F           52   D1 0004E         CMPL    STATUS, #2104              ; 2184
                                           03   12 00055         BNEQ    5$                         ;
                              52           01   D0 00057 4$:     MOVL    #1, STATUS                 ; 2185
                              1E           52   E8 0005A 5$:     BLBS    STATUS, 6$                 ; 2186
                00000870      8F           52   D1 0005D         CMPL    STATUS, #2160              ; 2187
                                           15   13 00064         BEQL    6$                         ;
                                           52   DD 00066         PUSHL   STATUS                     ; 2188
                                00000000'  EF   DD 00068         PUSHL   RWSV_SAVE_FAB              ;
                                00000000G  8F   DD 0006E         PUSHL   #BACKUP$_POSITERR          ;
                00000000G     00           03   FB 00074         CALLS   #3, FILE_ERROR             ;
```

```
                                50              52  D0 0007B 6$:    MOVL    STATUS, R0                              ; 2191
                                                    04 0007E        RET                                             ;

; Routine Size:  127 bytes,    Routine Base:  CODE + 0602
```

```
;1101        2192  1 %SBTTL 'READ_LABEL - read tape label'
;1102        2193  1 GLOBAL ROUTINE READ_LABEL (BUFFER, LABEL_TYPE) =
;1103        2194  1
;1104        2195  1 !++
;1105        2196  1 !
;1106        2197  1 ! FUNCTIONAL DESCRIPTION:
;1107        2198  1 !
;1108        2199  1 !       This routine reads and verifies a magtape label record.
;1109        2200  1 !
;1110        2201  1 ! CALLING SEQUENCE:
;1111        2202  1 !       READ_LABEL (BUFFER, LABEL_TYPE)
;1112        2203  1 !
;1113        2204  1 ! INPUT PARAMETERS:
;1114        2205  1 !       BUFFER: address of buffer to read label
;1115        2206  1 !       LABEL_TYPE: optional label type to check for
;1116        2207  1 !
;1117        2208  1 ! IMPLICIT INPUTS:
;1118        2209  1 !       NONE
;1119        2210  1 !
;1120        2211  1 ! OUTPUT PARAMETERS:
;1121        2212  1 !       NONE
;1122        2213  1 !
;1123        2214  1 ! IMPLICIT OUTPUTS:
;1124        2215  1 !       NONE
;1125        2216  1 !
;1126        2217  1 ! ROUTINE VALUE:
;1127        2218  1 !       TRUE if label is valid
;1128        2219  1 !       BACKUP$_NOTANSI if any checks fail
;1129        2220  1 !
;1130        2221  1 ! SIDE EFFECTS:
;1131        2222  1 !       NONE
;1132        2223  1 !
;1133        2224  1 !--
;1134        2225  1
;1135        2226  2 BEGIN
;1136        2227  2
;1137        2228  2 BUILTIN
;1138        2229  2         ACTUALCOUNT;
;1139        2230  2
;1140        2231  2 MAP
;1141        2232  2         BUFFER          : REF BBLOCK;   ! label buffer arg
;1142        2233  2
;1143        2234  2 LOCAL
;1144        2235  2         STATUS,                         ! system service status
;1145        2236  2         IO_STATUS       : VECTOR [4, WORD]; ! I/O status block
;1146        2237  2
;1147        2238  2 EXTERNAL ROUTINE
;1148        2239  2         FILE_ERROR;                     ! signal file related error
;1149        2240  2
;1150        2241  2 ! Read a record from the input channel and make the approriate checks.
;1151        2242  2 !
;1152        2243  2
;1153      P 2244  2 STATUS = $QIOW (CHAN = .RWSV_CHAN,
;1154      P 2245  2                 FUNC = IO$_READLBLK,
;1155      P 2246  2                 IOSB = IO_STATUS,
;1156      P 2247  2                 P1   = .BUFFER,
;1157      P 2248  2                 P2   = 90
```

```
;1158        2249  2                    );
;1159        2250  2 IF .STATUS THEN STATUS = .IO_STATUS[0];
;1160        2251  2 IF .STATUS EQL SS$_ENDOFTAPE
;1161        2252  2 THEN STATUS = TRUE;
;1162        2253  2 IF NOT .STATUS
;1163        2254  2 THEN
;1164        2255  3     BEGIN
;1165        2256  3     IF .STATUS EQL SS$_DATAOVERUN
;1166        2257  3     THEN RETURN BACKUP$_NOTANSI
;1167        2258  3     ELSE IF .STATUS EQL SS$_ENDOFFILE
;1168        2259  3     THEN RETURN SS$_ENDOFVOLUME
;1169        2260  3     ELSE
;1170        2261  3         IF .IO_STATUS[1] GTRU 80
;1171        2262  3         THEN RETURN .STATUS
;1172        2263  3         ELSE FILE_ERROR (BACKUP$_LABELERR, .RWSV_SAVE_FAB, .STATUS);
;1173        2264  2     END;
;1174        2265  2
;1175        2266  2 IF .IO_STATUS[1] NEQ 80
;1176        2267  2 THEN RETURN BACKUP$_NOTANSI;
;1177        2268  2
;1178        2269  2 IF ACTUALCOUNT () GEQU 2
;1179        2270  2 THEN
;1180        2271  3     BEGIN
;1181        2272  3     IF .BUFFER[HD1$L_HD1LID] NEQ .LABEL_TYPE
;1182        2273  3     THEN RETURN BACKUP$_NOTANSI;
;1183        2274  2     END;
;1184        2275  2
;1185        2276  2 TRUE
;1186        2277  1 END;                                    ! End of routine READ_LABEL


                                 0000 00000          .ENTRY  READ_LABEL, Save nothing                ;2193
                           5E    08   C2 00002          SUBL2   #8, SP
                                 7E   7C 00005          CLRQ    -(SP)                                   ;2249
                                 7E   7C 00007          CLRQ    -(SP)
                           7E     5A   8F 9A 00009          MOVZBL  #90, -(SP)
                                  04   AC DD 0000D          PUSHL   BUFFER
                                 7E   7C 00010          CLRQ    -(SP)
                                  20   AE 9F 00012          PUSHAB  IO_STATUS
                                       21 DD 00015          PUSHL   #33
                              00000000' EF DD 00017          PUSHL   RWSV_CHAN
                                 7E   D4 0001D          CLRL    -(SP)
00000000G                  00         0C FB 0001F          CALLS   #12, SYS$QIOW
                           03         50 E9 00026          BLBC    STATUS, 1$                              ;2250
                           50         6E 3C 00029          MOVZWL  IO_STATUS, STATUS
00000878                   8F         50 D1 0002C 1$:      CMPL    STATUS, #2168                           ;2251
                                      03 12 00033          BNEQ    2$
                           50         01 D0 00035          MOVL    #1, STATUS                              ;2252
                           35         50 E8 00038 2$:      BLBS    STATUS, 4$                              ;2253
00000838                   8F         50 D1 0003B          CMPL    STATUS, #2104                           ;2256
                                      40 13 00042          BEQL    5$
00000870                   8F         50 D1 00044          CMPL    STATUS, #2160                           ;2258
                                      06 12 0004B          BNEQ    3$
                           50     09A0 8F 3C 0004D          MOVZWL  #2464, R0                               ;2259
```

```
                                                 04 00052          RET
                                0050    8F       02    AE  B1 00053 3$:      CMPW    IO_STATUS+2, #80                        ;  2261
                                                       34  1A 00059          BGTRU   7$                                      ;
                                                       50  DD 0005B          PUSHL   STATUS                                  ;  2263
                                            00000000'  EF  DD 0005D          PUSHL   RWSV_SAVE_FAB                           ;
                                            00000000G  8F  DD 00063          PUSHL   #BACKUP$_LABELERR                       ;
                                00000000G   00             03  FB 00069     CALLS   #3, FILE_ERROR                          ;
                                    0050    8F       02    AE  B1 00070 4$:      CMPW    IO_STATUS+2, #80                        ;  2266
                                                       0C  12 00076          BNEQ    5$                                      ;
                                            02             6C  91 00078          CMPB    (AP), #2                                ;  2269
                                                       0F  1F 0007B          BLSSU   6$                                      ;
                                      08    AC       04    BC  D1 0007D          CMPL    @BUFFER, LABEL_TYPE                     ;  2272
                                                       08  13 00082          BEQL    6$                                      ;
                                            50 00000000G   8F  D0 00084 5$:      MOVL    #BACKUP$_NOTANSI, R0                    ;  2273
                                                           04  0008B          RET                                             ;
                                            50             01  D0 0008C 6$:      MOVL    #1, R0                                  ;  2277
                                                           04  0008F 7$:      RET                                             ;

; Routine Size:  144 bytes,    Routine Base:  CODE + 0681
```

```
;1188        2278  1 %SBTTL 'WRITE_TM - write tape mark'
;1189        2279  1 GLOBAL ROUTINE WRITE_TM : NOVALUE =
;1190        2280  1
;1191        2281  1 !++
;1192        2282  1 !
;1193        2283  1 ! FUNCTIONAL DESCRIPTION:
;1194        2284  1 !
;1195        2285  1 !       This routine writes a tape mark onto the current output tape.
;1196        2286  1 !
;1197        2287  1 ! CALLING SEQUENCE:
;1198        2288  1 !       WRITE_TM ()
;1199        2289  1 !
;1200        2290  1 ! INPUT PARAMETERS:
;1201        2291  1 !       NONE
;1202        2292  1 !
;1203        2293  1 ! IMPLICIT INPUTS:
;1204        2294  1 !       NONE
;1205        2295  1 !
;1206        2296  1 ! OUTPUT PARAMETERS:
;1207        2297  1 !       NONE
;1208        2298  1 !
;1209        2299  1 ! IMPLICIT OUTPUTS:
;1210        2300  1 !       NONE
;1211        2301  1 !
;1212        2302  1 ! ROUTINE VALUE:
;1213        2303  1 !       NONE
;1214        2304  1 !
;1215        2305  1 ! SIDE EFFECTS:
;1216        2306  1 !       NONE
;1217        2307  1 !
;1218        2308  1 !--
;1219        2309  1
;1220        2310  2 BEGIN
;1221        2311  2
;1222        2312  2 LOCAL
;1223        2313  2         STATUS,                         ! general status value
;1224        2314  2         IO_STATUS       : VECTOR [4, WORD]; ! I/O status block
;1225        2315  2
;1226        2316  2 EXTERNAL ROUTINE
;1227        2317  2         FILE_ERROR;                     ! signal file related error
;1228        2318  2
;1229      P 2319  2 STATUS = $QIOW (CHAN = .RWSV_CHAN,
;1230      P 2320  2                 FUNC = IO$_WRITEOF,
;1231      P 2321  2                 IOSB = IO_STATUS
;1232        2322  2                 );
;1233        2323  2 IF .STATUS THEN STATUS = .IO_STATUS[0];
;1234        2324  2 IF .STATUS EQL SS$_ENDOFTAPE
;1235        2325  2 THEN STATUS = TRUE;
;1236        2326  2 IF NOT .STATUS
;1237        2327  2 THEN FILE_ERROR (BACKUP$_LABELERR, .RWSV_SAVE_FAB, .STATUS);
;1238        2328  2
;1239        2329  1 END;                                     ! End of routine WRITE_TM
```

```
                                    0000 00000          .ENTRY  WRITE_TM, Save nothing                  ; 2279
                          5E        08   C2 00002          SUBL2   #8, SP
                                    7E   7C 00005          CLRQ    -(SP)                               ; 2322
                                    7E   7C 00007          CLRQ    -(SP)
                                    7E   7C 00009          CLRQ    -(SP)
                                    7E   7C 0000B          CLRQ    -(SP)
                              20    AE   9F 0000D          PUSHAB  IO_STATUS
                                    28   DD 00010          PUSHL   #40
                    00000000'       EF   DD 00012          PUSHL   RWSV_CHAN
                                    7E   D4 00018          CLRL    -(SP)
          00000000G 00              0C   FB 0001A          CALLS   #12, SYS$QIOW
                    03              50   E9 00021          BLBC    STATUS, 1$                          ; 2323
                    50              6E   3C 00024          MOVZWL  IO_STATUS, STATUS
          00000878  8F              50   D1 00027 1$:      CMPL    STATUS, #2168                       ; 2324
                                    03   12 0002E          BNEQ    2$
                    50              01   D0 00030          MOVL    #1, STATUS                          ; 2325
                    15              50   E8 00033 2$:      BLBS    STATUS, 3$                          ; 2326
                                    50   DD 00036          PUSHL   STATUS                              ; 2327
                    00000000'       EF   DD 00038          PUSHL   RWSV_SAVE_FAB
                    00000000G       8F   DD 0003E          PUSHL   #BACKUP$_LABELERR
          00000000G 00              03   FB 00044          CALLS   #3, FILE_ERROR
                                    04   0004B 3$:         RET                                         ; 2329

; Routine Size:  76 bytes,    Routine Base:  CODE + 0711
```

```
;  1241          2330  1 %SBTTL 'WRITE_LABEL - write tape label'
;  1242          2331  1 GLOBAL ROUTINE WRITE_LABEL (BUFFER) : NOVALUE =
;  1243          2332  1
;  1244          2333  1 !++
;  1245          2334  1 !
;  1246          2335  1 ! FUNCTIONAL DESCRIPTION:
;  1247          2336  1 !
;  1248          2337  1 !       This routine writes a label onto the current output tape.
;  1249          2338  1 !
;  1250          2339  1 ! CALLING SEQUENCE:
;  1251          2340  1 !       WRITE_LABEL (BUFFER)
;  1252          2341  1 !
;  1253          2342  1 ! INPUT PARAMETERS:
;  1254          2343  1 !       BUFFER: address of buffer containing label to be written
;  1255          2344  1 !
;  1256          2345  1 ! IMPLICIT INPUTS:
;  1257          2346  1 !       NONE
;  1258          2347  1 !
;  1259          2348  1 ! OUTPUT PARAMETERS:
;  1260          2349  1 !       NONE
;  1261          2350  1 !
;  1262          2351  1 ! IMPLICIT OUTPUTS:
;  1263          2352  1 !       NONE
;  1264          2353  1 !
;  1265          2354  1 ! ROUTINE VALUE:
;  1266          2355  1 !       NONE
;  1267          2356  1 !
;  1268          2357  1 ! SIDE EFFECTS:
;  1269          2358  1 !       NONE
;  1270          2359  1 !
;  1271          2360  1 !--
;  1272          2361  1
;  1273          2362  2 BEGIN
;  1274          2363  2
;  1275          2364  2 LOCAL
;  1276          2365  2         STATUS,                                 ! general status value
;  1277          2366  2         IO_STATUS       : VECTOR [4, WORD]; ! I/O status block
;  1278          2367  2
;  1279          2368  2 EXTERNAL ROUTINE
;  1280          2369  2         FILE_ERROR;                             ! signal file related error
;  1281          2370  2
;  1282        P 2371  2 STATUS = $QIOW (CHAN = .RWSV_CHAN,
;  1283        P 2372  2                 FUNC = IO$_WRITELBLK,
;  1284        P 2373  2                 IOSB = IO_STATUS,
;  1285        P 2374  2                 P1   = .BUFFER,
;  1286        P 2375  2                 P2   = 80
;  1287          2376  2                 );
;  1288          2377  2 IF .STATUS THEN STATUS = .IO_STATUS[0];
;  1289          2378  2 IF .STATUS EQL SS$_ENDOFTAPE
;  1290          2379  2 THEN STATUS = TRUE;
;  1291          2380  2 IF NOT .STATUS
;  1292          2381  2 THEN FILE_ERROR (BACKUP$_LABELERR, .RWSV_SAVE_FAB, .STATUS);
;  1293          2382  2
;  1294          2383  1 END;                                            ! End of routine WRITE_LABEL
```

```
                                    0000 00000          .ENTRY  WRITE_LABEL, Save nothing                       ; 2331
                          5E        08   C2 00002          SUBL2   #8, SP                                          ;
                                    7E   7C 00005          CLRQ    -(SP)                                           ; 2376
                                    7E   7C 00007          CLRQ    -(SP)                                           ;
                          7E        8F   9A 00009          MOVZBL  #80, -(SP)                                      ;
                                    04   AC DD 0000D       PUSHL   BUFFER                                          ;
                                    7E   7C 00010          CLRQ    -(SP)                                           ;
                                    20   AE 9F 00012       PUSHAB  IO_STATUS                                       ;
                                    20   DD 00015          PUSHL   #32                                             ;
                              00000000' EF   DD 00017       PUSHL   RWSV_CHAN                                       ;
                                    7E   D4 0001D          CLRL    -(SP)                                           ;
                00000000G     00    0C   FB 0001F          CALLS   #12, SYS$QIOW                                   ;
                              03    50   E9 00026          BLBC    STATUS, 1$                                      ; 2377
                              50    6E   3C 00029          MOVZWL  IO_STATUS, STATUS                               ;
                00000878      8F    50   D1 0002C 1$:     CMPL    STATUS, #2168                                   ; 2378
                                    03   12 00033          BNEQ    2$                                              ;
                              50    01   D0 00035          MOVL    #1, STATUS                                      ; 2379
                              15    50   E8 00038 2$:     BLBS    STATUS, 3$                                      ; 2380
                                    50   DD 0003B          PUSHL   STATUS                                          ; 2381
                              00000000' EF   DD 0003D       PUSHL   RWSV_SAVE_FAB                                   ;
                              00000000G 8F   DD 00043       PUSHL   #BACKUP$_LABELERR                               ;
                00000000G     00    03   FB 00049          CALLS   #3, FILE_ERROR                                  ;
                                    04   00050 3$:        RET                                                     ; 2383

; Routine Size:  81 bytes,    Routine Base:  CODE + 075D
```

```
;1296    2384  1 %SBTTL 'JULIAN_DATE - generate Julian date in tape label'
;1297    2385  1 ROUTINE JULIAN_DATE (BUFFER) : NOVALUE =
;1298    2386  1
;1299    2387  1 !++
;1300    2388  1 !
;1301    2389  1 ! FUNCTIONAL DESCRIPTION:
;1302    2390  1 !
;1303    2391  1 !       This routine places todays date into the specified 6 byte
;1304    2392  1 !       buffer in ANSI Julian date format.
;1305    2393  1 !
;1306    2394  1 ! CALLING SEQUENCE:
;1307    2395  1 !       JULIAN_DATE (BUFFER)
;1308    2396  1 !
;1309    2397  1 ! INPUT PARAMETERS:
;1310    2398  1 !       NONE
;1311    2399  1 !
;1312    2400  1 ! IMPLICIT INPUTS:
;1313    2401  1 !       NONE
;1314    2402  1 !
;1315    2403  1 ! OUTPUT PARAMETERS:
;1316    2404  1 !       BUFFER: buffer into which to place date
;1317    2405  1 !
;1318    2406  1 ! IMPLICIT OUTPUTS:
;1319    2407  1 !       NONE
;1320    2408  1 !
;1321    2409  1 ! ROUTINE VALUE:
;1322    2410  1 !       NONE
;1323    2411  1 !
;1324    2412  1 ! SIDE EFFECTS:
;1325    2413  1 !       NONE
;1326    2414  1 !
;1327    2415  1 !--
;1328    2416  1
;1329    2417  2 BEGIN
;1330    2418  2
;1331    2419  2 MAP
;1332    2420  2         BUFFER          : REF VECTOR [,BYTE];
;1333    2421  2
;1334    2422  2 BIND
;1335    2423  2         DAYTBL = UPLIT WORD(0,31,59,90,120,151,181,212,243,273,304,334,365)
;1336    2424  2                                         : VECTOR [, WORD];
;1337    2425  2
;1338    2426  2 LITERAL
;1339    2427  2         N_YEAR          = 0,            ! year in time buffer
;1340    2428  2         N_MONTH         = 1,            ! month in buffer
;1341    2429  2         N_DAY           = 2;            ! day in buffer
;1342    2430  2
;1343    2431  2 LOCAL
;1344    2432  2         TIME_BUFFER     : VECTOR [7, WORD], ! buffer to receive system time
;1345    2433  2         DAY,                            ! day of year
;1346    2434  2         STRING_BUFFER   : VECTOR [7, BYTE], ! FAO output string
;1347    2435  2         STRING_DESC     : VECTOR [2];   ! descriptor for above
;1348    2436  2
;1349    2437  2 ! Get the system time in numerical format. Then run the month through
;1350    2438  2 ! the table to compute day in year, adjusting for leap year. (Note
;1351    2439  2 ! we handle only the 4 year leap year cycle. The Julian date format
;1352    2440  2 ! will have long crumbled to ashes by the time we see the next 100
```

```
;  1353          2441  2 ! year cycle.)
;  1354          2442  2 !
;  1355          2443  2
;  1356          2444  2 $NUMTIM (TIMBUF = TIME_BUFFER);
;  1357          2445  2 DAY = .TIME_BUFFER[N_DAY] + .DAYTBL[.TIME_BUFFER[N_MONTH]-1];
;  1358          2446  2 IF .(TIME_BUFFER[N_YEAR])<0,2> EQL 0
;  1359          2447  2 AND .TIME_BUFFER[N_MONTH] GTRU 2
;  1360          2448  2 THEN DAY = .DAY + 1;
;  1361          2449  2
;  1362          2450  2 ! Convert to string and put it in the buffer.
;  1363          2451  2 !
;  1364          2452  2
;  1365          2453  2 STRING_DESC[0] = 7;
;  1366          2454  2 STRING_DESC[1] = STRING_BUFFER;
;  1367        P 2455  2 $FAO   ($DESCRIPTOR ('!4ZL!3ZL'),
;  1368        P 2456  2          0,
;  1369        P 2457  2          STRING_DESC[0],
;  1370        P 2458  2          .TIME_BUFFER[N_YEAR],
;  1371        P 2459  2          .DAY
;  1372          2460  2          );
;  1373          2461  2 BUFFER[0] = ' ';
;  1374          2462  2 CH$MOVE (5, STRING_BUFFER[2], BUFFER[1]);
;  1375          2463  2
;  1376          2464  1 END;                                    ! End of routine JULIAN_DATE


0111  00F3  00D4  00B5  0097  0078  005A  003B  001F  0000  007AE P.AAD:  .WORD   0, 31, 59, 90, 120, 151, 181, 212, 243, -   ;
                                  016D  014E  0130  007C2                273, 304, 334, 365                                ;
                      4C  5A  33  21  4C  5A  34  21  007C8 P.AAF:  .ASCII  \!4ZL!3ZL\                                    ;
                                    00000008  007D0 P.AAE:  .LONG   8                                                     ;
                                    00000000' 007D4         .ADDRESS P.AAF                                                ;

                                              DAYTBL=               P.AAD
                                                    .EXTRN  SYS$NUMTIM, SYS$FAO

                                    003C 00000 JULIAN_DATE:
                                               .WORD    Save R2,R3,R4,R5                                                  ; 2385
                  5E                   20  C2 00002          SUBL2    #32, SP                                             ;
                                       7E  D4 00005          CLRL     -(SP)                                               ; 2444
                              14       AE  9F 00007          PUSHAB   TIME_BUFFER                                         ;
        00000000G  00                  02  FB 0000A          CALLS    #2, SYS$NUMTIM                                      ;
                  50                12  AE  3C 00011          MOVZWL   TIME_BUFFER+2, R0                                   ; 2445
                  51                14  AE  3C 00015          MOVZWL   TIME_BUFFER+4, R1                                   ;
                  50             B7 AF40  3C 00019          MOVZWL   DAYTBL-2[R0], DAY                                   ;
                  50                51  C0 0001E          ADDL2    R1, DAY                                                 ;
                  03                10  AE  93 00021          BITB     TIME_BUFFER, #3                                     ; 2446
                                       08  12 00025          BNEQ     1$                                                  ;
                  02                12  AE  B1 00027          CMPW     TIME_BUFFER+2, #2                                   ; 2447
                                       02  1B 0002B          BLEQU    1$                                                  ;
                                       50  D6 0002D          INCL     DAY                                                 ; 2448
                  6E                   07  D0 0002F 1$:      MOVL     #7, STRING_DESC                                     ; 2453
              04  AE                08  AE  9E 00032          MOVAB    STRING_BUFFER, STRING_DESC+4                        ; 2454
                                       50  DD 00037          PUSHL    DAY                                                 ; 2460
                  7E                   14  AE  3C 00039          MOVZWL   TIME_BUFFER, -(SP)                                  ;
                                       08  AE  9F 0003D          PUSHAB   STRING_DESC                                         ;
                                       7E  D4 00040          CLRL     -(SP)                                               ;
```

```
                                    B3   AF   9F 00042          PUSHAB  P.AAE                                                    ;
                          00000000G 00        05   FB 00045    CALLS   #5, SYS$FAO                                            ;
                                    50   04   AC   D0 0004C    MOVL    BUFFER, R0                                             ; 2461
                                    60        20   90 00050    MOVB    #32, (R0)                                              ;
                                01  A0   0A   AE   05   28 00053    MOVC3   #5, STRING_BUFFER+2, 1(R0)                         ; 2462
                                                   04 00059    RET                                                            ; 2464

; Routine Size:  90 bytes,    Routine Base:  CODE + 07D8
```

```
;1378        2465  1 %SBTTL 'FORMAT_VOLOWNER - format tape volume owner'
;1379        2466  1 ROUTINE FORMAT_VOLOWNER (VOL_LABEL, OWNER, PROTECTION) : NOVALUE  =
;1380        2467  1
;1381        2468  1 !++
;1382        2469  1 !
;1383        2470  1 ! FUNCTIONAL DESCRIPTION:
;1384        2471  1 !
;1385        2472  1 !       This routine formats the volume owner filed in VOL1
;1386        2473  1 !
;1387        2474  1 ! CALLING SEQUENCE:
;1388        2475  1 !       FORMAT_VOLOWNER (VOL_LABEL, OWNER, PROTECTION)
;1389        2476  1 !
;1390        2477  1 ! INPUT PARAMETERS:
;1391        2478  1 !       VOL_LABEL - address of VOL1 label
;1392        2479  1 !       OWNER - owner of tape
;1393        2480  1 !       PROTECTION - tape protection
;1394        2481  1 !
;1395        2482  1 ! IMPLICIT INPUTS:
;1396        2483  1 !       D%C preinitialized
;1397        2484  1 !
;1398        2485  1 ! OUTPUT PARAMETERS:
;1399        2486  1 !       NONE
;1400        2487  1 !
;1401        2488  1 ! IMPLICIT OUTPUTS:
;1402        2489  1 !       NONE
;1403        2490  1 !
;1404        2491  1 ! ROUTINE VALUE:
;1405        2492  1 !       NONE
;1406        2493  1 !
;1407        2494  1 ! SIDE EFFECTS:
;1408        2495  1 !       NONE
;1409        2496  1 !
;1410        2497  1 ! USER ERRORS:
;1411        2498  1 !       NONE
;1412        2499  1 !
;1413        2500  1 !--
;1414        2501  1
;1415        2502  2 BEGIN
;1416        2503  2
;1417        2504  2 MAP
;1418        2505  2         VOL_LABEL       : REF BBLOCK,   ! address of VOL1 label
;1419        2506  2         PROTECTION      : BITVECTOR;    ! protection to be encoded on tape
;1420        2507  2
;1421        2508  2 LOCAL
;1422        2509  2         DESCR           : VECTOR [2],   ! descriptor
;1423        2510  2         P;                              ! pointer
;1424        2511  2
;1425        2512  2 LITERAL
;1426        2513  2         WORLD_WRITE = 13,
;1427        2514  2         WORLD_READ = 12,
;1428        2515  2         GROUP_WRITE = 9,
;1429        2516  2         GROUP_READ = 8;
;1430        2517  2
;1431        2518  2
;1432        2519  2 ! First convert binary owner to ASCII
;1433        2520  2 !
;1434        2521  2
```

```
;  1435          2522  2 DESCR[0] = 10;
;  1436          2523  2 DESCR[1] = VOL_LABEL[VL1$T_VOLOWNER] + 3;
;  1437        P 2524  2 $FAO (
;  1438        P 2525  2         $DESCRIPTOR('!5OW!5OW'),
;  1439        P 2526  2         0,
;  1440        P 2527  2         DESCR[0],
;  1441          2528  2         .OWNER<16,16>,.OWNER<0,16>);
;  1442          2529  2
;  1443          2530  2 ! Now format protection
;  1444          2531  2 !
;  1445          2532  2
;  1446          2533  2 IF NOT .PROTECTION[GROUP_READ] OR NOT .PROTECTION[WORLD_READ]
;  1447          2534  2 THEN
;  1448          2535  3     BEGIN
;  1449          2536  3     P = VOL_LABEL[VL1$T_VOLOWNER] + 8;
;  1450          2537  3     (.P)<0,8> = .(.P)<0,8> + ('A' - '0');
;  1451          2538  2     END;
;  1452          2539  2
;  1453          2540  2 ! Now if group can also write, blank fill member field
;  1454          2541  2 !
;  1455          2542  2
;  1456          2543  2 IF NOT .PROTECTION[GROUP_WRITE]
;  1457          2544  2 THEN CH$FILL(' ',5,VOL_LABEL[VL1$T_VOLOWNER] + 8);
;  1458          2545  2
;  1459          2546  2 IF NOT .PROTECTION[WORLD_READ]
;  1460          2547  2 THEN
;  1461          2548  3     BEGIN
;  1462          2549  3     P = VOL_LABEL[VL1$T_VOLOWNER] + 3;
;  1463          2550  3     (.P)<0,8> = .(.P)<0,8> + ('A' - '0');
;  1464          2551  2     END;
;  1465          2552  2
;  1466          2553  2 IF NOT .PROTECTION[WORLD_WRITE]
;  1467          2554  2 THEN CH$FILL(' ',10,VOL_LABEL[VL1$T_VOLOWNER] + 3);
;  1468          2555  2
;  1469          2556  1 END;                                  ! end of routine FORMAT_VOLOWNER


                                                00832 P.AAH:  .ASCII  \!5OW!5OW\                        :
                                                0083A         .BLKB   2
                                    00000008    0083C P.AAG:  .LONG   8                                 ;
                                    00000000'   00840         .ADDRESS P.AAH                          ;


                                        00FC 00000 FORMAT_VOLOWNER:
                                                   .WORD   Save R2,R3,R4,R5,R6,R7                       ; 2466
                                    5E          04  C2 00002         SUBL2   #4, SP                      ;
                                                0A  DD 00005         PUSHL   #10                         ; 2522
                                    57      04  AC  D0 00007         MOVL    VOL_LABEL, R7               ; 2523
                                04  AE      28  A7  9E 0000B         MOVAB   40(R7), DESCR+4             ;
                                    7E      08  AC  3C 00010         MOVZWL  OWNER, -(SP)                ; 2528
                                    7E      0A  AC  3C 00014         MOVZWL  OWNER+2, -(SP)              ;
                                            08  AE  9F 00018         PUSHAB  DESCR                       ;
                                                7E  D4 0001B         CLRL    -(SP)                       ;
                                            D8  AF  9F 0001D         PUSHAB  P.AAG                       ;
                                00000000G   00      05  FB 00020    CALLS   #5, SYS$FAO                 ;
```

```
                                05      0D   AC  E9 00027          BLBC    PROTECTION+1, 1$                        ; 2533
                        07      0D   AC      04  E0 0002B          BBS     #4, PROTECTION+1, 2$                    ;
                                56      2D   A7  9E 00030 1$:      MOVAB   45(R7), P                               ; 2536
                                66           11  80 00034          ADDB2   #17, (P)                                ; 2537
                        07      0D   AC      01  E0 00037 2$:      BB      #1, PROTECTION+1, 3$                    ; 2543
                05      20           6E      00  2C 0003C          MOVC5   #0, (SP), #32, #5, 45(R7)               ; 2544
                                          2D   A7      00041                                                      ;
                        07      0D   AC      04  E0 00043 3$:      BBS     #4, PROTECTION+1, 4$                    ; 2546
                                56      28   A7  9E 00048          MOVAB   40(R7), P                               ; 2549
                                66           11  80 0004C          ADDB2   #17, (P)                                ; 2550
                        07      0D   AC      05  E0 0004F 4$:      BBS     #5, PROTECTION+1, 5$                    ; 2553
                0A      20           6E      00  2C 00054          MOVC5   #0, (SP), #32, #10, 40(R7)              ; 2554
                                          28   A7      00059                                                      ;
                                                04  0005B 5$:      RET                                             ; 2556

; Routine Size:  92 bytes,    Routine Base:  CODE + 0844
```

```
;  1471          2557  1 %SBTTL 'MAKE_VOL1 - format VOL1 header label'
;  1472          2558  1 GLOBAL ROUTINE MAKE_VOL1 (BUFFER) : NOVALUE =
;  1473          2559  1
;  1474          2560  1 !++
;  1475          2561  1 !
;  1476          2562  1 ! FUNCTIONAL DESCRIPTION:
;  1477          2563  1 !
;  1478          2564  1 !       This routine formats a tape volume header label in the
;  1479          2565  1 !       indicated buffer.
;  1480          2566  1 !
;  1481          2567  1 ! CALLING SEQUENCE:
;  1462          2568  1 !       MAKE_VOL1 (BUFFER)
;  1483          2569  1 !
;  1484          2570  1 ! INPUT PARAMETERS:
;  1485          2571  1 !       NONE
;  1486          2572  1 !
;  1487          2573  1 ! IMPLICIT INPUTS:
;  1488          2574  1 !       NONE
;  1489          2575  1 !
;  1490          2576  1 ! OUTPUT PARAMETERS:
;  1491          2577  1 !       BUFFER: buffer to write label into
;  1492          2578  1 !
;  1493          2579  1 ! IMPLICIT OUTPUTS:
;  1494          2580  1 !       NONE
;  1495          2581  1 !
;  1496          2582  1 ! ROUTINE VALUE:
;  1497          2583  1 !       NONE
;  1498          2584  1 !
;  1499          2585  1 ! SIDE EFFECTS:
;  1500          2586  1 !       NONE
;  1501          2587  1 !
;  1502          2588  1 !--
;  1503          2589  1
;  1504          2590  2 BEGIN
;  1505          2591  2
;  1506          2592  2 MAP
;  1507          2593  2         BUFFER          : REF BBLOCK;   ! label buffer arg
;  1508          2594  2
;  1509          2595  2 LOCAL
;  1510          2596  2         P               : REF BBLOCK,   ! structure pointer
;  1511          2597  2         DESCRIPTOR      : VECTOR [2];   ! string descriptor for FAO
;  1512          2598  2
;  1513          2599  2 ! Initialize the label buffer and set up the basic volume label.
;  1514          2600  2 !
;  1515          2601  2
;  1516          2602  2 CH$FILL (' ', 80, .BUFFER);
;  1517          2603  2 BUFFER[VL1$L_VL1LID] = 'VOL1';
;  1518          2604  2 CH$COPY(
;  1519          2605  2     .BBLOCK[RWSV_SAVE_FAB[FC_NAM], NAM$B_NAME],
;  1520          2606  2     .BBLOCK[RWSV_SAVE_FAB[FC_NAM], NAM$L_NAME],
;  1521          2607  2     %C' ',
;  1522          2608  2     VL1$S_VOLLBL, BUFFER[VL1$T_VOLLBL]);
;  1523          2609  2
;  1524          2610  2 ! If an explicit label was specified, get it. Use the segment number
;  1525          2611  2 ! to pick the right entry from the label list. If we are out of list,
;  1526          2612  2 ! use the first entry and append the reel number.
;  1527          2613  2 !
```

```
;  1528          2614  2
;  1529          2615  2 P = 0;
;  1530          2616  2 IF .QUAL[QUAL_LABE]
;  1531          2617  2 THEN
;  1532          2618  3     BEGIN
;  1533          2619  3     P = .QUAL[QUAL_LABE_LIST];
;  1534          2620  3     DECR J FROM .RWSV_VOL_NUMBER-1 TO 1
;  1535          2621  3     DO
;  1536          2622  4         BEGIN
;  1537          2623  4         P = .P[QUAL_NEXT];
;  1538          2624  4         IF .P EQL 0 THEN EXITLOOP;
;  1539          2625  3         END;
;  1540          2626  3     IF .P NEQ 0
;  1541          2627  3     THEN CH$MOVE (VL1$S_VOLLBL, P[QUAL_LABE_VALUE], BUFFER[VL1$T_VOLLBL])
;  1542          2628  3     ELSE CH$MOVE (VL1$S_VOLLBL, BBLOCK[.QUAL[QUAL_LABE_LIST], QUAL_LABE_VALUE], BUFFER[VL1$T_VOLLBL]);
;  1543          2629  2     END;
;  1544          2630  2
;  1545          2631  2 ! Append the save set volume number to the volume label if this is a
;  1546          2632  2 ! continuation volume and an explicit label was not available.
;  1547          2633  2 !
;  1548          2634  2
;  1549          2635  2 IF .RWSV_VOL_NUMBER GTRU 1
;  1550          2636  2 AND .P EQL 0
;  1551          2637  2 THEN
;  1552          2638  3     BEGIN
;  1553          2639  3     INCR J FROM 0 TO VL1$S_VOLLBL-2-1
;  1554          2640  3     DO
;  1555          2641  4         BEGIN
;  1556          2642  4         IF .VECTOR [BUFFER[VL1$T_VOLLBL], .J; VL1$S_VOLLBL, BYTE] EQL ' '
;  1557          2643  4         THEN VECTOR [BUFFER[VL1$T_VOLLBL], .J; VL1$S_VOLLBL, BYTE] = '_';
;  1558          2644  3         END;
;  1559          2645  3     DESCRIPTOR[0] = 2;
;  1560          2646  3     DESCRIPTOR[1] = BUFFER[VL1$T_VOLLBL] + VL1$S_VOLLBL - 2;
;  1561        P 2647  3     $FAO ($DESCRIPTOR ('!2ZL'),
;  1562        P 2648  3           0,
;  1563        P 2649  3           DESCRIPTOR[0],
;  1564        P 2650  3           .RWSV_VOL_NUMBER
;  1565          2651  3           );
;  1566          2652  2     END;
;  1567          2653  2
;  1568          2654  2 ! For the first volume, save the volume label as the file set ID.
;  1569          2655  2 !
;  1570          2656  2
;  1571          2657  2 IF .RWSV_VOL_NUMBER LEQU 1
;  1572          2658  2 THEN CH$MOVE (HD1$S_FILESETID, BUFFER[VL1$T_VOLLBL], RWSV_FILESET_ID);
;  1573          2659  2
;  1574          2660  2 ! Fill in the remaining fixed fields.
;  1575          2661  2 !
;  1576          2662  2
;  1577          2663  2 (BUFFER[VL1$T_VOLOWNER])<0,24> = 'D%C';
;  1578          2664  2 BUFFER[VL1$B_DECSTDVER] = '1';
;  1579          2665  2 BUFFER[VL1$B_LBLSTDVER] = '3';
;  1580          2666  2
;  1581          2667  2 ! If ownership and protection are specified, fill in the fields in
;  1582          2668  2 ! the label.
;  1583          2669  2 !
;  1584          2670  2
```

```
;1585        2671  2 IF .QUAL[QUAL_PROT]
;1586        2672  2 THEN FORMAT_VOLOWNER (.BUFFER,
;1587        2673  2                       IF .QUAL[QUAL_O_OWN_UIC] THEN .QUAL[QUAL_O_OWN_VALU] ELSE .JPI_UIC,
;1588        2674  2                       .QUAL[QUAL_PROT_VALUE]);
;1589        2675  2
;1590        2676  1 END;                                    ! End of routine MAKE_VOL1


                                    4C  5A  32  21  008A0 P.AAJ:  .ASCII  \!2ZL\                                    ;
                                          00000004  008A4 P.AAI:  .LONG   4                                         ;
                                          00000000' 008A8         .ADDRESS P.AAJ                                    ;


                                          03FC 00000        .ENTRY  MAKE_VOL1, Save R2,R3,R4,R5,R6,R7,R8,R9     ; 2558
                              59 00000000'  EF   9E 00002   MOVAB   RWSV_VOL_NUMBER, R9                          ;
                                 5E         08   C2 00009   SUBL2   #8, SP                                       ;
                                 57   04    AC   D0 0000C   MOVL    BUFFER, R7                                   ; 2602
    0050  8F         20          6E         00   2C 00010   MOVC5   #0, (SP), #32, #80, (R7)                    ;
                                           67        00017                                                       ;
                              67 314C4F56   8F   D0 00018   MOVL    #827084630, (R7)                             ; 2603
                                 50   0C    A9   D0 0001F   MOVL    RWSV_SAVE_FAB, R0                            ; 2605
                                 51   00CF  C0   9A 00023   MOVZBL  207(R0), R1                                  ;
                                 58   04    A7   9E 00028   MOVAB   4(R7), R8                                    ; 2608
           06        20    00E0  D0         51   2C 0002C   MOVC5   R1, @224(R0), #32, #6, (R8)                 ;
                                           68        00033                                                       ;
                                           56   D4 00034   CLRL    P                                            ; 2615
                                      6A   A9   95 00036   TSTB    QUAL+10                                      ; 2616
                                           25   18 00039   BGEQ    5$                                           ;
                                    50     00B0  C9   D0 0003B   MOVL    QUAL+80, R0                            ; 2619
                                    56           50   D0 00040   MOVL    R0, P                                  ;
                                    51           69   3C 00043   MOVZWL  RWSV_VOL_NUMBER, J                      ; 2620
                                                 05   11 00046   BRB     2$                                     ;
                                    56           66   D0 00048 1$:  MOVL    (P), P                               ; 2623
                                                 03   13 0004B   BEQL    3$                                     ; 2624
                                    F8           51   F5 0004D 2$:  SOBGTR  J, 1$                               ; 2620
                                                 56   D5 00050 3$:  TSTL    P                                    ; 2626
                                                 07   13 00052   BEQL    4$                                     ;
    68         04    A6                        06   28 00054   MOVC3   #6, 4(P), (R8)                          ; 2627
                                                 05   11 00059   BRB     5$                                     ;
    68         04    A0                        06   28 0005B 4$:  MOVC3   #6, 4(R0), (R8)                      ; 2628
                     01                        69   B1 00060 5$:  CMPW    RWSV_VOL_NUMBER, #1                  ; 2635
                                                 30   1B 00063   BLEQU   8$                                     ;
                                                 56   D5 00065   TSTL    P                                      ; 2636
                                                 2C   12 00067   BNEQ    8$                                     ;
                                                 50   D4 00069   CLRL    J                                      ; 2639
                     20                        6048 91 0006B 6$:  CMPB    (J)[R8], #32                         ; 2642
                                                 05   12 0006F   BNEQ    7$                                     ;
                               6048        5F   8F   90 00071   MOVB    #95, (J)[R8]                           ; 2643
    F1                 50                     03   F3 00076 7$:  AOBLEQ  #3, J, 6$                             ; 2639
                       6E                     C2   D0 0007A   MOVL    #2, DESCRIPTOR                         ; 2645
                  04   AE         08         A7   9E 0007D   MOVAB   8(R7), DESCRIPTOR+4                       ; 2646
                       7E                     69   3C 00082   MOVZWL  RWSV_VOL_NUMBER, -(SP)                   ; 2651
                                        04     AE   9F 00085   PUSHAB  DESCRIPTOR                               ;
                                               7E   D4 00088   CLRL    -(SP)                                  ;
                                          FF6A CF   9F 0008A   PUSHAB  P.AAI                                    ;
```

```
                        00000000G  00              04  FB 0008E          CALLS   #4, SYS$FAO                                    ;
                                   01              69  B1 00095 8$:      CMPW    RWSV_VOL_NUMBER, #1                            ; 2657
                                                   05  1A 00098          BGTRU   9$                                             ;
                     EC    A9                      68  28 0009A          MOVC3   #6, (R8), RWSV_FILESET_ID                      ; 2658
          25   A7              18                 00 00432544  8F  F0 0009F 9$:  INSV    #4400452, #0, #24, 37(R7)                      ; 2663
                                       32         A7  31  90 000A9          MOVB    #49, 50(R7)                                    ; 2664
                                       4F         A7  33  90 000AD          MOVB    #51, 79(R7)                                    ; 2665
                              1B       6E         A9  03  E1 000B1          BBC     #3, QUAL+14, 12$                               ; 2671
                                                  7E       00B4  C9  3C 000B6          MOVZWL  QUAL+84, -(SP)                         ; 2674
                              06       6C         A9  04  E1 000BB          BBC     #4, QUAL+12, 10$                               ; 2673
                                                       00A0  C9  DD 000C0          PUSHL   QUAL+64                                ;
                                                             04  11 000C4          BRB     11$                                    ;
                                                       FF34  C9  DD 000C6 10$:     PUSHL   JPI_UIC                                ;
                                                             57  DD 000CA 11$:     PUSHL   R7                                     ; 2672
                                    FEC7   CF              03  FB 000CC          CALLS   #3, FORMAT_VOLOWNER                    ;
                                                             04 000D1 12$:      RET                                            ; 2676

; Routine Size:  210 bytes,    Routine Base:  CODE + 08AC
```

```
;  1592          2677  1 %SBTTL 'MAKE_HDR1 - format HDR1 header label'
;  1593          2678  1 GLOBAL ROUTINE MAKE_HDR1 (BUFFER) : NOVALUE =
;  1594          2679  1
;  1595          2680  1 !++
;  1596          2681  1 !
;  1597          2682  1 ! FUNCTIONAL DESCRIPTION:
;  1598          2683  1 !
;  1599          2684  1 !       This routine formats a tape file header label 1 in the
;  1600          2685  1 !       indicated buffer.
;  1601          2686  1 !
;  1602          2687  1 ! CALLING SEQUENCE:
;  1603          2688  1 !       MAKE_HDR1 (BUFFER)
;  1604          2689  1 !
;  1605          2690  1 ! INPUT PARAMETERS:
;  1606          2691  1 !       NONE
;  1607          2692  1 !
;  1608          2693  1 ! IMPLICIT INPUTS:
;  1609          2694  1 !       NONE
;  1610          2695  1 !
;  1611          2696  1 ! OUTPUT PARAMETERS:
;  1612          2697  1 !       BUFFER: buffer to write label into
;  1613          2698  1 !
;  1614          2699  1 ! IMPLICIT OUTPUTS:
;  1615          2700  1 !       NONE
;  1616          2701  1 !
;  1617          2702  1 ! ROUTINE VALUE:
;  1618          2703  1 !       NONE
;  1619          2704  1 !
;  1620          2705  1 ! SIDE EFFECTS:
;  1621          2706  1 !       NONE
;  1622          2707  1 !
;  1623          2708  1 !--
;  1624          2709  1
;  1625          2710  2 BEGIN
;  1626          2711  2
;  1627          2712  2 MAP
;  1628          2713  2         BUFFER          : REF BBLOCK;   ! label buffer arg
;  1629          2714  2
;  1630          2715  2 BIND
;  1631          2716  2         PROTO_HDR1      = UPLIT BYTE ('000100 00000 00000 000000DECVMSBACKUP        ');
;  1632          2717  2
;  1633          2718  2 LOCAL
;  1634          2719  2         DESCRIPTOR      : VECTOR [2];   ! string descriptor for FAO
;  1635          2720  2
;  1636          2721  2 BUFFER[HD1$L_HD1LID] = 'HDR1';
;  1637          2722  2 CH$COPY (.COM_SSNAME[DSC$W_LENGTH],
;  1638          2723  2         .COM_SSNAME[DSC$A_POINTER],
;  1639          2724  2         ' '
;  1640          2725  2         HD1$S_FILEID,
;  1641          2726  2         BUFFER[HD1$T_FILEID]);
;  1642          2727  2 CH$MOVE (HD1$S_FILESETID, RWSV_FILESET_ID, BUFFER[HD1$T_FILESETID]);
;  1643          2728  2 CH$MOVE (80-$BYTEOFFSET (HD1$T_GENNO), PROTO_HDR1, BUFFER[HD1$T_GENNO]);
;  1644          2729  2
;  1645          2730  2 ! Generate file section number, sequence number, and creation date.
;  1646          2731  2 !
;  1647          2732  2
;  1648          2733  2 DESCRIPTOR[0] = 4;
```

```
;  1649          2734  2 DESCRIPTOR[1] = BUFFER[HD1$T_FILESECNO];
;  1650        P 2735  2 $FAO   ($DESCRIPTOR ('!4ZL'),
;  1651        P 2736  2         0,
;  1652        P 2737  2         DESCRIPTOR[0],
;  1653        P 2738  2         .RWSV_VOL_NUMBER
;  1654          2739  2         );
;  1655          2740  2 DESCRIPTOR[1] = BUFFER[HD1$T_FILESEQNO];
;  1656        P 2741  2 $FAO   ($DESCRIPTOR ('!4ZL'),
;  1657        P 2742  2         0,
;  1658        P 2743  2         DESCRIPTOR[0],
;  1659        P 2744  2         .RWSV_FILE_NUMBER
;  1660          2745  2         );
;  1661          2746  2 JULIAN_DATE (BUFFER[HD1$T_CREATEDT]);
;  1662          2747  2
;  1663          2748  1 END;                                          ! End of routine MAKE_HDR1


30  30  20  30  30  30  30  30  20  30  30  31  30  30  30  0097E P.AAK:  .ASCII  \000100 00000 00000 000000DECVMSBACKUP   \ ;
4D  56  43  45  44  30  30  30  30  30  3C  20  30  30  30  0098D                                                            ;
                    20  20  20  50  55  4B  43  41  42  53  0099C                                                            ;
                                        20  20  20  20  20  009A6         .ASCII  \     \                                  ;
                                            4C  5A  34  21  009AB P.AAM:  .ASCII  \!4ZL\                                  ;
                                                            009AF         .BLKB   1                                        ;
                                              00000004      009B0 P.AAL:  .LONG   4                                        ;
                                              00000000'     009B4         .ADDRESS P.AAM                                   ;
                                            4C  5A  34  21  009B8 P.AAO:  .ASCII  \!4ZL\                                  ;
                                              00000004      009BC P.AAN:  .LONG   4                                        ;
                                              00000000'     009C0         .ADDRESS P.AAO                                   ;

                                                          PROTO_HDR1=       P.AAK


                                                    03FC  00000         .ENTRY  MAKE_HDR1, Save R2,R3,R4,R5,R6,R7,R8,R9     ; 2678
                                          59 00000000G  00   9E  00002  MOVAB   SYS$FAO, R9                                  ;
                                          58        AE  AF   9E  00009  MOVAB   PROTO_HDR1, R8                               ;
                                          57 00000000'  EF   9E  0000D  MOVAB   COM_SSNAME, R7                             ;
                                          5E            08   C2  00014  SUBL2   #8, SP                                     ;
                                          56        04  AC   D0  00017  MOVL    BUFFER, R6                                  ; 2721
                                          66 31524448   8F   D0  0001B  MOVL    #827475016, (R6)                            ;
11              20        04  B7                  67   2C  00022  MOVC5   COM_SSNAME, @COM_SSNAME+4, #32, #17, 4(R6)  ; 2726
                                                04  A6        00028                                                        ;
          15    A6      FF1C  C7                  06   28  0002A  MOVC3   #6, RWSV_FILESET_ID, 21(R6)                   ; 2727
          23    A6            68                  2D   28  00031  MOVC3   #45, PROTO_HDR1, 35(R6)                       ; 2728
                              6E                  04   D0  00036  MOVL    #4, DESCRIPTOR                                ; 2733
                        04    AE          1B      A6   9E  00039  MOVAB   27(R6), DESCRIPTOR+4                         ; 2734
                              7E        FF30      C7   3C  0003E  MOVZWL  RWSV_VOL_NUMBER, -(SP)                        ; 2739
                                          04      AE   9F  00043  PUSHAB  DESCRIPTOR                                    ;
                                                  7E   D4  00046  CLRL    -(SP)                                         ;
                                          32      A8   9F  00048  PUSHAB  P.AAL                                         ;
                                        69        04   FB  0004B  CALLS   #4, SYS$FAO                                   ;
                        04    AE          1F      A6   9E  0004E  MOVAB   31(R6), DESCRIPTOR+4                         ; 2740
                                      FF34      C7   DD  00053  PUSHL   RWSV_FILE_NUMBER                              ; 2745
                                          04      AE   9F  00057  PUSHAB  DESCRIPTOR                                    ;
                                                  7E   D4  0005A  CLRL    -(SP)                                         ;
                                          3E      A8   9F  0005C  PUSHAB  P.AAN                                         ;
                                        69        04   FB  0005F  CALLS   #4, SYS$FAO                                   ;
```

```
                                 29   A6  9F 00062         PUSHAB  41(R6)                                  ; 2746
                      FDAA   CF          01  FB 00065         CALLS   #1, JULIAN_DATE                        ;
                                         04 0006A         RET                                            ; 2748

; Routine Size:  107 bytes,    Routine Base:  CODE + 09C4
```

```
;1665           2749  1 %SBTTL 'MAKE_HDR2 - format HDR2 header label'
;1666           2750  1 GLOBAL ROUTINE MAKE_HDR2 (BUFFER) : NOVALUE =
;1667           2751  1
;1668           2752  1 !++
;1669           2753  1 !
;1670           2754  1 ! FUNCTIONAL DESCRIPTION:
;1671           2755  1 !
;1672           2756  1 !       This routine formats a tape file header label 2 in the
;1673           2757  1 !       indicated buffer.
;1674           2758  1 !
;1675           2759  1 ! CALLING SEQUENCE:
;1676           2760  1 !       MAKE_HDR2 (BUFFER)
;1677           2761  1 !
;1678           2762  1 ! INPUT PARAMETERS:
;1679           2763  1 !       NONE
;1680           2764  1 !
;1681           2765  1 ! IMPLICIT INPUTS:
;1682           2766  1 !       NONE
;1683           2767  1 !
;1684           2768  1 ! OUTPUT PARAMETERS:
;1685           2769  1 !       BUFFER: buffer to write label into
;1686           2770  1 !
;1687           2771  1 ! IMPLICIT OUTPUTS:
;1688           2772  1 !       NONE
;1689           2773  1 !
;1690           2774  1 ! ROUTINE VALUE:
;1691           2775  1 !       NONE
;1692           2776  1 !
;1693           2777  1 ! SIDE EFFECTS:
;1694           2778  1 !       NONE
;1695           2779  1 !
;1696           2780  1 !--
;1697           2781  1
;1698           2782  2 BEGIN
;1699           2783  2
;1700           2784  2 MAP
;1701           2785  2         BUFFER          : REF BBLOCK;   . label buffer arg
;1702           2786  2
;1703           2787  2 LOCAL
;1704           2788  2         DESCRIPTOR      : VECTOR [2];   ! FAO string descriptor
;1705           2789  2
;1706           2790  2
;1707           2791  2 CH$FILL (' ', 80, .BUFFER);
;1708           2792  2 BUFFER[HD2$L_HD2LID] = 'HDR2';
;1709           2793  2 BUFFER[HD2$B_RECFORMAT] = 'F';
;1710           2794  2 DESCRIPTOR[0] = HD2$S_BLOCKLEN + HD2$S_RECLEN;
;1711           2795  2 DESCRIPTOR[1] = BUFFER[HD2$T_BLOCKLEN];
;1712         P 2796  2 $FAO    ($DESCRIPTOR ('!2(5ZL)'),
;1713         P 2797  2         0,
;1714         P 2798  2         DESCRIPTOR[0],
;1715         P 2799  2         .QUAL[QUAL_BLOC_VALUE],
;1716         P 2800  2         .QUAL[QUAL_BLOC_VALUE]
;1717           2801  2         );
;1718           2802  2
;1719           2803  2 BUFFER[HD2$B_FORMCNTRL] = 'M';
;1720           2804  2 BUFFER[HD2$T_BUFOFF] = '00';
;1721           2805  2
```

```
; 1722          2806  1 END;                                    ! End of routine MAKE_HDR2


                                                    29  4C  5A  35  28  32  21  00A2F P.AAQ:  .ASCII  \!2(5ZL)\                ;
                                                                                00A36          .BLKB   2                          ;
                                                                00000007        00A38 P.AAP:  .LONG   7                          ;
                                                                00000000'       00A3C          .ADDRESS P.AAQ                    ;


                                                                            00FC 00000         .ENTRY  MAKE_HDR2, Save R2,R3,R4,R5,R6,R7  ; 2750
                                                            57 00000000'    EF   9E 00002      MOVAB   QUAL+72, R7                ;
                                                            5E              08   C2 00009      SUBL2   #8, SP                     ;
                                                            56        04    AC   D0 0000C      MOVL    BUFFER, R6                 ; 2791
0050    8F              20                                  6E              00   2C 00010      MOVC5   #0, (SP), #32, #80, (R6)   ;
                                                                            66   00017        ;
                                                            66 32524448     8F   D0 00018      MOVL    #844252232, (R6)           ; 2792
                                                      04    A6        46    8F   90 0001F      MOVB    #70, 4(R6)                 ; 2793
                                                            6E              0A   D0 00024      MOVL    #10, DESCRIPTOR            ; 2794
                                                      04    AE        05    A6   9E 00027      MOVAB   5(R6), DESCRIPTOR+4        ; 2795
                                                            7E              67   3C 0002C      MOVZWL  QUAL+72, -(SP)             ; 2801
                                                            7E              67   3C 0002F      MOVZWL  QUAL+72, -(SP)             ;
                                                                      08    AE   9F 00032      PUSHAB  DESCRIPTOR                 ;
                                                                            7E   D4 00035      CLRL    -(SP)                      ;
                                                                      BE    AF   9F 00037      PUSHAB  P.AAP                      ;
                                                          00000000G   00          05   FB 0003A      CALLS   #5, SYS$FAO                ;
                                                                24    A6        4D   8F   90 00041      MOVB    #77, 36(R6)                ; 2803
                                                                32    A6      3030   8F   B0 00046      MOVW    #12336, 50(R6)             ; 2804
                                                                                04   0004C      RET                                ; 2806

; Routine Size:  77 bytes,    Routine Base:  CODE + 0A40


; 1723          2807  1
; 1724          2808  1 END
; 1725          2809  0 ELUDOM



                                                                                               .EXTRN  LIB$SIGNAL

;                                       PSECT SUMMARY
;
;       Name                            Bytes                           Attributes
;
;  COMMON                                 2124  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  OVR,NOPIC,ALIGN(2)
;  CODE                                   2701  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        88         0       1000          00:01.7



;                                       COMMAND QUALIFIERS

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

; Size:         2495 code + 2330 data bytes
; Run Time:        00:54.7
; Elapsed Time:    03:21.2
; Lines/CPU Min:    3078
; Lexemes/CPU-Min: 30074
; Memory Used:  315 pages
; Compilation Complete
```

TAPEUTIL
LIS

STAINVOL
LIS

UTILITY
LIS

STATOSYS
LIS

STAPARSE
LIS