
MMM MMM TTTTTTTTTTTTTTT AAAAAAAAA AAAAAAAAA CCCCCCCCCCCC PPPPPPPPPPPP
MMM MMM TTTTTTTTTTTTTTT AAAAAAAAA AAAAAAAAA CCCCCCCCCCCC PPPPPPPPPPPP
MMM MMM TTTTTTTTTTTTTTT AAAAAAAAA AAAAAAAAA CCCCCCCCCCCC PPPPPPPPPPPP
MMMMMM MMMMMM TTT AAA AAA AAA AAA CCC PPP PPP
MMMMMM MMMMMM TTT AAA AAA AAA AAA CCC PPP PPP
MMMMMM MMMMMM TTT AAA AAA AAA AAA CCC PPP PPP
MMM MMM MMM TTT AAA AAA AAA AAA CCC PPP PPP
MMM MMM MMM TTT AAA AAA AAA AAA CCC PPP PPP
MMM MMM MMM TTT AAA AAA AAA AAA CCC PPP PPP
MMM MMM TTT AAA AAA AAA AAA CCC PPPPPPPPPPPP
MMM MMM TTT AAA AAA AAA AAA CCC PPPPPPPPPPPP
MMM MMM TTT AAA AAA AAA AAA CCC PPPPPPPPPPPP
MMM MMM TTT AAAAAAAAAAAAAAA AAAAAAAAAAAAAAA CCC PPP
MMM MMM TTT AAAAAAAAAAAAAAA AAAAAAAAAAAAAAA CCC PPP
MMM MMM TTT AAAAAAAAAAAAAAA AAAAAAAAAAAAAAA CCC PPP
MMM MMM TTT AAA AAA AAA AAA CCC PPP
MMM MMM TTT AAA AAA AAA AAA CCC PPP
MMM MMM TTT AAA AAA AAA AAA CCC PPP
MMM MMM TTT AAA AAA AAA AAA CCCCCCCCCCCC PPP
MMM MMM TTT AAA AAA AAA AAA CCCCCCCCCCCC PPP
MMM MMM TTT AAA AAA AAA AAA CCCCCCCCCCCC PPP

FILEID**RDATTR

```
RRRRRRRR    DDDDDDDD      AAAAAA    TTTTTTTTTT  TTTTTTTTTT  RRRRRRRR
RRRRRRRR    DDDDDDDD      AAAAAA    TTTTTTTTTT  TTTTTTTTTT  RRRRRRRR
RR      RR  DD      DD   AA      AA      TT          TT      RR      RR
RR      RR  DD      DD   AA      AA      TT          TT      RR      RR
RR      RR  DD      DD   AA      AA      TT          TT      RR      RR
RR      RR  DD      DD   AA      AA      TT          TT      RR      RR
RRRRRRRR    DD      DD   AA      AA      TT          TT      RRRRRRRR
RRRRRRRR    DD      DD   AA      AA      TT          TT      RRRRRRRR
RR  RR      DD      DD   AAAAAAAAAA      TT          TT      RR  RR
RR  RR      DD      DD   AAAAAAAAAA      TT          TT      RR  RR
RR    RR    DD      DD   AA      AA      TT          TT      RR    RR        ....
RR    RR    DD      DD   AA      AA      TT          TT      RR    RR        ....
RR      RR  DDDDDDDD     AA      AA      TT          TT      RR      RR      ....
RR      RR  DDDDDDDD     AA      AA      TT          TT      RR      RR      ....

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
;     2          0002  0 MODULE RDATTR (LANGUAGE (BLISS32) ,
;     3          0003  0                IDENT = 'V04-000'
;     4          0004  0                ) =
;     5          0005  1 BEGIN
;     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:  MTAACP
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:
;    35          0035  1 !       This module reads attributes
;    36          0036  1 !
;    37          0037  1 ! ENVIRONMENT:
;    38          0038  1 !
;    39          0039  1 !       STARLET operating system, including privileged system services
;    40          0040  1 !       and internal exec routines.
;    41          0041  1 !
;    42          0042  1 !--
;    43          0043  1
;    44          0044  1 !
;    45          0045  1 !
;    46          0046  1 ! AUTHOR:  D. H. GILLESPIE,      CREATION DATE: 19-MAY-77 13:55
;    47          0047  1 !
;    48          0048  1 ! MODIFIED BY:
;    49          0049  1 !
;    50          0050  1 !       V03-013 ACG0415         Andrew C. Goldstein,    9-Apr-1984  16:27
;    51          0051  1 !               Fix probing of access mode ATR's
;    52          0052  1 !
;    53          0053  1 !       V03-012 LMP0221         L. Mark Pilant,         28-Mar-1984  14:53
;    54          0054  1 !               Change UCB$L_OWNUIC to ORB$L_OWNER and UCB$W_VPROT to
;    55          0055  1 !               ORB$W_PROT.
;    56          0056  1 !
;    57          0057  1 !       V03-011 ACG0410         Andrew C. Goldstein,    23-Mar-1984  13:09
```

```
;    58          0058  1 !                Implement access mode attribute for buffer probing
;    59          0059  1 !
;    60          0060  1 !        V03-010 MMD0268         Meg Dumont,     23-Mar-1984  9:14
;    61          0061  1 !                Fix long file name support such that for ANSI version
;    62          0062  1 !                3 volumes it converts the exentsion length to
;    63          0063  1 !                ASCII before writing it to the label.
;    64          0064  1 !
;    65          0065  1 !        V03-009 MMD0249         Meg Dumont,     27-Feb-1984  17:35
;    66          0066  1 !                Fix to the buffer offset to suppor.
;    67          0067  1 !
;    68          0068  1 !        V03-008 MMD0232         Meg Dumont,     3-Feb-1984  16:32
;    69          0069  1 !                Fix bug where attribute in range was tested against MAX_CODE-1.
;    70          0070  1 !
;    71          0071  1 !        V03-007 MMD0214         Meg Dumont,     3-Jan-1984  14:37
;    72          0072  1 !                Add support for Buffer Offset on file open for read access only.
;    73          0073  1 !
;    74          0074  1 !        V03-006 MMD0194         Meg Dumont,     4-Aug-1983  10:54
;    75          0075  1 !                Fix the atribute ATR$_JOURNAL to zero user buffer
;    76          0076  1 !
;    77          0077  1 !        V03-005 MMD0163         Meg Dumont,     26-Apr-1983  9:41
;    78          0078  1 !                Change references to 80 to the symbol ANSI_LBLSZ. Add long
;    79          0079  1 !                file name support which includes changing the routine
;    80          0080  1 !                ANSI_FILE_NAME to understand that the file name is split
;    81          0081  1 !                between the HDR1 and HDR4 labels.
;    82          0082  1 !
;    83          0083  1 !        V03-004 MMD0090         Meg Dumont
;    84          0084  1 !                Take out assume on atr.
;    85          0085  1 !
;    86          0086  1 !        V03-003 MMD0129         Meg Dumont,     8-Apr-1983  14:21
;    87          0087  1 !                Change to atribute processing so that MTAACP ignores any ACCL's
;    88          0088  1 !
;    89          0089  1 !        V03-002 MMD0002         Meg Dumont,     12-Nov-1982  10:53
;    90          0090  1 !                Take out ASSUME on the number of ATR.
;    91          0091  1 !
;    92          0092  1 !        V03-001 MMD0001         Meg Dumont,     6-Jul-1982  17:44
;    93          0093  1 !                Fix ASSUME for ATR$C_MAX_CODE from 30 to 37.
;    94          0094  1 !
;    95          0095  1 !        V02-012 DMW00065        David Michael Walp      8-Jan-1982
;    96          0096  1 !                Added support for HDR1 access code
;    97          0097  1 !
;    98          0098  1 !        V02-011 DMW00051        David Michael Walp      10-Nov-1981
;    99          0099  1 !                Added support for ANSI 17 "a" character name to be passed
;   100          0100  1 !                back through ASCNAME attribute.
;   101          0101  1 !
;   102          0102  1 !        V02-010 DMW00048        David Michael Walp      3-Nov-1981
;   103          0103  1 !                added Journal ATTRIBUTE support
;   104          0104  1 !
;   105          0105  1 !        V02-009 DMW00045        David Michael Walp      28-Oct-1981
;   106          0106  1 !                Reformat module into routines, and comment out old
;   107          0107  1 !                dead code
;   108          0108  1 !
;   109          0109  1 !        V02-008 DMW00041        David Michael Walp      2-Oct-1981
;   110          0110  1 !                Handle zero Julian date, and added zero time
;   111          0111  1 !
;   112          0112  1 !        V02-007 DMW0003         David Michael Walp      11-Nov-1980
;   113          0113  1 !                New BLISS compiler, FUNCTION declaration changed from
;   114          0114  1 !                BBLOCK TO BLOCK.  Old compiler used to give a longword
```

```
;  115     0115  1 !               with a declaration of "BBLOCK [1]".
;  116     0116  1 !
;  117     0117  1 !       V02-006 REFORMAT        Maria del C. Nasr       30-Jun-1980
;  118     0118  1 !
;  119     0119  1 !       A0005   MCN0006         Maria del C. Nasr       08-Nov-1979   19:10
;  120     0120  1 !               Change size of file characteris attribute.
;  121     0121  1 !
;  122     0122  1 !       A0004   MCN0003         Maria del C. Nasr       16-Oct-1979   14:19
;  123     0123  1 !               Add HDR3 processing
;  124     0124  1 !
;  125     0125  1 !       A0003   MCN0004         Maria del C. Nasr       15-Oct-1979   15:36
;  126     0126  1 !               Changed to use new file header structur name
;  127     0127  1 !
;  128     0128  1 !**
;  129     0129  1
;  130     0130  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;  131     0131  1
;  132     0132  1 REQUIRE 'SRC$:MTADEF.B32';
;  133     0516  1
;  134     0517  1 FORWARD ROUTINE
;  135     0518  1     ANSI_FILE_NAME      : COMMON_CALL NOVALUE,  ! return the full ANSI name
;  136     0519  1     CALCULATE_STATISTICS: COMMON_CALL NOVALUE,  ! dummy up a stat block
;  137     0520  1     CONVERT_DATE        :             NOVALUE,  ! converts Julian to binnary
;  138     0521  1     GET_BLOCK_SIZE      :             NOVALUE,  ! get the block size
;  139     0522  1     GET_BUFFER_OFFSET   :             NOVALUE,  ! get the buffer offset length
;  140     0523  1     HANDLER             : COMMON_CALL NOVALUE,  ! exception handler
;  141     0524  1     MOVE_DATA           : COMMON_CALL NOVALUE,  ! move attribute date
;  142     0525  1     READ_ATTRIBUTE      : COMMON_CALL NOVALUE,  ! read attributes
;  143     0526  1     SET_BCNT            : COMMON_CALL NOVALUE;  ! set # of valid buffer descr
;  144     0527  1
;  145     0528  1 EXTERNAL
;  146     0529  1     CURRENT_UCB         : REF BBLOCK,           ! address of current UCB
;  147     0530  1     HDR1                : REF BBLOCK,           ! address of HDR1(EOF1) label
;  148     0531  1     HDR4                : REF BBLOCK,           ! address of HDR4(EOF4) label
;  149     0532  1     IO_PACKET           : REF BBLOCK;           ! address of current IRP
;  150     0533  1
;  151     0534  1 EXTERNAL ROUTINE
;  152     0535  1     FORMAT_F11HOD1      : COMMON_CALL,          ! format & write ODS1 header
;  153     0536  1     LIB$CVT_DTB         : ADDRESSING_MODE (ABSOLUTE);! convert decimal 2 binary
;  154     0537  1
;  155     0538  1 OWN
;  156     0539  1     CURRENT_ATTRIB;                             ! current attribute number
;  157     0540  1
;  158     0541  1
;  159     0542  1     ! work area for file header                                              ; F
;  160     0543  1     !
;  161     0544  1 PSECT  GLOBAL = $DATA$;
;  162     0545  1 GLOBAL DATA             : BLOCK [512, BYTE],
;  163     0546  1        ODS1_HEADER      : BLOCK [512, BYTE];
```

```
;  165          0547  1 !++
;  166          0548  1 !
;  167          0549  1 ! The Attribute Table is indexed by attribute number.  Each entry is a
;  168          0550  1 ! longword which contains a description of possible file attributes.
;  169          0551  1 ! The entries on the table are:  maximum attribute size, byte offset of
;  170          0552  1 ! attribute into structure, index of the action routine to process the
;  171          0553  1 ! attribute, and a description flag.
;  172          0554  1 !
;  173          0555  1 !--
;  174          0556  1
;  175          0557  1     ! read attributes actior routine encoding
;  176          0558  1     !
;  177          0559  1     LITERAL
;  178          0560  1         ATR_HDR         = 0,                    ! copy from header
;  179          0561  1         ATR_STAT        = 1,                    ! build statistics block
;  180          0562  1         ATR_BLSZ        = 2,                    ! block size
;  181          0563  1         ATR_USERLBL     = 3,                    ! user labels
;  182          0564  1         ATR_NOP         = 4,                    ! ignore attribute
;  183          0565  1         ATR_ENDLBLAST   = 5,                    ! end of file user labels
;  184          0566  1         ATR_ZERO        = 6,                    ! zero valued attribute
;  185          0567  1         ATR_BINDATE     = 7,                    ! date and time
;  186          0568  1         ATR_UIC         = 8,                    ! owner uic
;  187          0569  1         ATR_ANSI_NAME   = 9,                    ! ANSI name in ASCII Name Attr
;  188          0570  1         ATR_HDR1_ACC    = 10,                   ! HDR1 access character
;  189          0571  1         ATR_BUFFER_OFFSET = 11,                 ! Buffer offset length is returned
;  190          0572  1         ATR_ACMODE      = 12;                   ! buffer access mode
;  191          0573  1
;  192          0574  1     ! Attribute table flag field
;  193          0575  1
;  194          0576  1     LITERAL
;  195          0577  1         M_ATR_ACL       = 2;                    ! File header access control list area
;  196          0578  1     ! Check that number of attributes has not changed, because
;  197          0579  1     ! if it has then the value must be added to the table below
;  198          0580  1     ! else the Mtaacp will treat the ATR as a valid atribute even
;  199          0581  1     ! though it may not be.
;  200          0582  1
;  201          0583  1 !    ASSUME ( 40, ATR$C_MAX_CODE );
;  202          0584  1
;  203          0585  1     ! the table and its values
;  204          0586  1     !
;  205          0587  1     GLOBAL      ATR_TABLE  : BLOCKVECTOR [ ATR$C_MAX_CODE, 1 ] INITIAL ( BYTE (
;  206          0588  1
;  207          0589  1 5,                  $BYTEOFFSET(FH1$W_FILEOWNER),  ATR_HDR,       0,
;  208          0590  1 3,                  $BYTEOFFSET(FH1$W_FILEPROT),   ATR_HDR,       0,
;  209          0591  1 ATR$S_UCHAR,        $BYTEOFFSET(FH1$W_FILECHAR),   ATR_HDR,       0,
;  210          0592  1 ATR$S_RECATTR,      $BYTEOFFSET(FH1$W_RECATTR),    ATR_HDR,       0,
;  211          0593  1 ATR$S_FILNAM
;  212          0594  1   +ATR$S_FILTYP
;  213          0595  1   +ATR$C_FILVER,   $BYTEOFFSET(FI1$W_FILENAME)
;  214          0596  1                     +FH1$K_LENGTH,                ATR_HDR,       0,
;  215          0597  1 ATR$S_FILTYP
;  216          0598  1   +ATR$S_FILVER,  $BYTEOFFSET(FI1$W_FILETYPE)
;  217          0599  1                     +FH1$K_LENGTH,                ATR_HDR,       0,
;  218          0600  1 ATR$S_FILVER,      $BYTEOFFSET(FI1$W_VERSION)
;  219          0601  1                     +FH1$K_LENGTH,                ATR_HDR,       0,
;  220          0602  1 ATR$S_EXPDAT,      $BYTEOFFSET(FI1$T_EXPDATE)
;  221          0603  1                     +FH1$K_LENGTH,                ATR_HDR,       0,
```

```
;  222          0604  1 ATR$S_STATBLK,   0,                                  ATR_STAT,      0,
;  223          0605  1
;  224          0606  1 ! the real size is 512 but it will not fit in a byte, so..
;  225          0607  1 !  there are hacks to test if size is zero and uses 512 when
;  226          0608  1 !  doing checks for attribute length
;  227          0609  1 !
;  228          0610  1 0,                 0,                                  ATR_HDR,       0,
;  229          0611  1 ATR$S_BLOCKSIZE, 0,                                    ATR_BLSZ,      0,
;  230          0612  1 ATR$S_USERLABEL, 0,                                    ATR_USERLBL,   0,
;  231          0613  1 ATR$S_ASCDATES,  $BYTEOFFSET(FI1$W_REVISION)
;  232          0614  1                   +FH1$K_LENGTH,                       ATR_HDR,       0,
;  233          0615  1 ATR$S_ALCONTROL, 0,                                    ATR_NOP,       0,
;  234          0616  1 ATR$S_ENDLBLAST, 0,                                    ATR_ENDLBLAST, 0,
;  235          0617  1 ATR$S_ASCNAME,   0,                                    ATR_ANSI_NAME, 0,
;  236          0618  1 ATR$S_CREDATE,   0,                                    ATR_BINDATE,   0,
;  237          0619  1 ATR$S_REVDATE,   0,                                    ATR_ZERO,      0,
;  238          0620  1 ATR$S_EXPDATE,   0,                                    ATR_BINDATE,   0,
;  239          0621  1 ATR$S_BAKDATE,   0,                                    ATR_ZERO,      0,
;  240          0622  1 ATR$S_UIC,       0,                                    ATR_UIC,       0,
;  241          0623  1 ATR$S_FPRO,      $BYTEOFFSET(FH1$W_FILEPROT)
;  242          0624  1                   +FH1$K_LENGTH,                       ATR_HDR,       0,
;  243          0625  1 ATR$S_RPRO,      0,                                    ATR_ZERO,      0,
;  244          0626  1 ATR$S_ACLEVEL,   0,                                    ATR_ZERO,      0,
;  245          0627  1 ATR$S_SEMASK,    0,                                    ATR_ZERO,      0,
;  246          0628  1 ATR$S_UIC_RO,    0,                                    ATR_UIC,       0,
;  247          0629  1 ATR$S_DIRSEQ,    0,                                    ATR_NOP,       0,
;  248          0630  1 ATR$S_BACKLINK,  0,                                    ATR_NOP,       0,
;  249          0631  1 ATR$S_JOURNAL,   0,                                    ATR_ZERO,      0,
;  250          0632  1 ATR$S_HDR1_ACC,  0,                                    ATR_HDR1_ACC,  0,
;  251          0633  1 !
;  252          0634  1 ! Use a hack similar to the 512 hack to get the max size of these fields
;  253          0635  1 !
;  254          0636  1 0,                 0,                                  ATR_ZERO,      M_ATR_ACL,
;  255          0637  1 0,                 0,                                  ATR_ZERO,      M_ATR_ACL,
;  256          0638  1 0,                 0,                                  ATR_ZERO,      M_ATR_ACL,
;  257          0639  1 0,                 0,                                  ATR_ZERO,      M_ATR_ACL,
;  258          0640  1 0,                 0,                                  ATR_ZERO,      M_ATR_ACL,
;  259          0641  1 0,                 0,                                  ATR_ZERO,      M_ATR_ACL,
;  260          0642  1 0,                 0,                                  ATR_ZERO,      0,
;  261          0643  1 ATR$S_ACLLENGTH, 0,                                    ATR_ZERO,      0,
;  262          0644  1 0,                 0,                                  ATR_ZERO,      M_ATR_ACL,
;  263          0645  1 0,                 0,                                  ATR_ZERO,      M_ATR_ACL,
;  264          0646  1 ATR$S_HIGHWATER, 0,                                    ATR_ZERO,      0,
;  265          0647  1 ATR$S_DUMMY_0,   0,                                    ATR_ZERO,      0,
;  266          0648  1 ATR$S_PRIVS_USED, 0,                                   ATR_ZERO,      0,
;  267          0649  1 0,                 0,                                  ATR_ZERO,      M_ATR_ACL,
;  268          0650  1 ATR$S_ACCESS_MODE, 0,                                  ATR_ACMODE,    0,
;  269          0651  1 0,                 0,                                  ATR_ZERO,      0,
;  270          0652  1 ATR$S_CLASS_MASK, 0,                                   ATR_ZERO,      0,
;  271          0653  1 ATR$S_BUFFER_OFFSET, 0,                                ATR_BUFFER_OFFSET, 0));
;  272          0654  1
```

```
;  274          0655  1 GLOBAL ROUTINE READ_ATTRIBUTE (ABD) : COMMON_CALL NOVALUE =
;  275          0656  1
;  276          0657  1 !++
;  277          0658  1 !
;  278          0659  1 ! FUNCTIONAL DESCRIPTION:
;  279          0660  1 !       This routine reads attributes
;  280          0661  1 !
;  281          0662  1 ! CALLING SEQUENCE:
;  282          0663  1 !       READ_ATTRIBUTE(ARG1)
;  283          0664  1 !
;  284          0665  1 ! INPUT PARAMETERS:
;  285          0666  1 !       ARG1 - address of buffer descriptors
;  286          0667  1 !
;  287          0668  1 ! IMPLICIT INPUTS:
;  288          0669  1 !       none
;  289          0670  1 !
;  290          0671  1 ! OUTPUT PARAMETERS:
;  291          0672  1 !       ARG1 - address of buffer descriptors
;  292          0673  1 !
;  293          0674  1 ! IMPLICIT OUTPUTS:
;  294          0675  1 !       none
;  295          0676  1 !
;  296          0677  1 ! ROUTINE VALUE:
;  297          0678  1 !       none
;  298          0679  1 !
;  299          0680  1 ! SIDE EFFECTS:
;  300          0681  1 !       Attribute data written into buffer packet, write-back enabled
;  301          0682  1 !
;  302          0683  1 !--
;  303          0684  1
;  304          0685  2     BEGIN
;  305          0686  2
;  306          0687  2     EXTERNAL REGISTER
;  307          0688  2         COMMON_REG;
;  308          0689  2
;  309          0690  2     LOCAL
;  310          0691  2         CODE,                                   ! current attribute code
;  311          0692  2         COUNT,                                  ! length of attribute buffer
;  312          0693  2         P,                                      ! address of current attribute
;  313          0694  2         ORB             : REF BBLOCK,           ! ORB address
;  314          0695  2         ACCESS_MODE,                            ! access mode to set for attribute buffer
;  315          0696  2         VALID_HEADER    : BITVECTOR [ 1 ];      ! has the ODS header created
;  316          0697  2
;  317          0698  2     MAP ABD             : REF BBLOCKVECTOR [, ABD$C_LENGTH];
;  318          0699  2                                                 ! addr of buffer desc
;  319          0700  2
;  320          0701  2     ! attribute table layout
;  321          0702  2     !
;  322          0703  2     MACRO   MAX  = 0,0,8,0%;                    ! maximum size of attribute
;  323          0704  2     MACRO   DISP = 0,8,8,0%;                    ! displacement into file header
;  324          0705  2     MACRO   ATR  = 0,16,8,0%;                   ! attribute type
;  325          0706  2     MACRO   FLAGS = 0,24,8,0%;                  ! attribute flags field
;  326          0707  2
;  327          0708  2     BUILTIN FP;
;  328          0709  2
;  329          0710  2
;  330          0711  2     ! initialize header variables
```

```
;  331          0712  2      !
;  332          0713  2      VALID_HEADER [0] = FALSE;
;  333          0714  2      ACCESS_MODE = .IO_PACKET[IRP$V_MODE];
;  334          0715  2
;  335          0716  2      .FP = HANDLER;
;  336          0717  2
;  337          0718  2      ! scan through buffer attribute descriptors filling them in one at a time
;  338          0719  2      !
;  339          0720  2      INCRU I FROM ABD$C_ATTRIB TO .IO_PACKET[IRP$W_BCNT] - 1 DO
;  340          0721  3          BEGIN
;  341          0722  3          CURRENT_ATTRIB = .I;                    ! for use by exception handler
;  342          0723  3          P              = .ABD [ .I, ABD$W_TEXT ] + ABD [ .I, ABD$W_TEXT ];
;  343          0724  3          COUNT          = .ABD [ .I, ABD$W_COUNT ];
;  344          0725  3          CODE           = .(.P)<0, 8> - 1;
;  345          0726  3
;  346          0727  3          ! if an error is found, set the descriptor count equal to the one
;  347          0728  3          ! that is in error this also inhibits the return of unprocessed
;  348          0729  3          ! descriptors.
;  349          0730  3          !
;  350          0731  3          ! check attribute code to see if its in range
;  351          0732  3          !
;  352          0733  3          IF .CODE GTRU ATR$C_MAX_CODE THEN ERR_EXIT(SS$_BADATTRIB);
;  353          0734  3
;  354          0735  3          ! check the size against limit
;  355          0736  3          !
;  356          0737  4          BEGIN
;  357          0738  4          LOCAL MAX_SIZE;
;  358          0739  5          MAX_SIZE = (IF .ATR_TABLE [.CODE, FLAGS] NEQ M_ATR_ACL
;  359          0740  5                      THEN .ATR_TABLE [.CODE, MAX]
;  360          0741  4                      ELSE 380);
;  361          0742  4          IF .MAX_SIZE EQL 0 THEN MAX_SIZE = 512;
;  362          0743  4          IF .COUNT GTRU .MAX_SIZE THEN ERR_EXIT(SS$_BADATTRIB);
;  363          0744  3          END;
;  364          0745  3
;  365          0746  3          ! now call subroutine to format attribute
;  366          0747  3          !
;  367          0748  3          CASE .ATR_TABLE [ .CODE, ATR ] FROM ATR_HDR TO ATR_ACMODE OF
;  368          0749  3              SET
;  369          0750  3              [ATR_HDR]        :  F_NOT .VALID_HEADER [0]
;  370          0751  3                                 .HEN
;  371          0752  4                                     BEGIN
;  372          0753  4                                     FORMAT_F11HOD1 (ODS1_HEADER);
;  373          0754  4                                     VALID_HEADER [0] = TRUE;
;  374          0755  3                                     END;
;  375          0756  3              [ATR_STAT]       : CALCULATE_STATISTICS ();
;  376          0757  3              [ATR_BLSZ]       : GET_BLOCK_SIZE ();
;  377          0758  3              [ATR_USERLBL]    : ERR_EXIT(SS$_BADATTRIB);
;  378          0759  3              [ATR_NOP]        : COUNT = 0;
;  379          0760  3              [ATR_ENDLBLAST]  : ERR_EXIT(SS$_BADATTRIB);
;  380          0761  3              [ATR_ZERO]       : CH$FILL(0, .COUNT, DATA);
;  381          0762  3              [ATR_BINDATE]    : CONVERT_DATE (.CODE);
;  382          0763  4              [ATR_UIC]        : BEGIN
;  383          0764  4                                 ORB = .CURRENT_UCB[UCB$L_ORB];
;  384          0765  4                                 DATA = .ORB[ORB$L_OWNER];
;  385          0766  3                                 END;
;  386          0767  3              [ATR_ANSI_NAME]  : ANSI_FILE_NAME ( COUNT );
;  387          0768  3              [ATR_HDR1_ACC]   : DATA [ 0, 0, 8, 0 ] = .HDR1 [ HD1$B_FILACCESS ];
```

```
;  388          0769  3              [ATR_BUFFER_OFFSET] : GET_BUFFER_OFFSET();
;  389          0770  4              [ATR_ACMODE]    : BEGIN
;  390          0771  4                                ACCESS_MODE = MAXU (.IO_PACKET[IRP$V_MODE], .(.P+1)<0,8>);
;  391          0772  4                                COUNT = 0;
;  392          0773  3                                END;
;  393          0774  3
;  394          0775  3              TES;
;  395          0776  3
;  396          0777  3          IF .ATR_TABLE [ .CODE, ATR ] EQL ATR_HDR
;  397        P 0778  3          THEN KERNEL_CALL(MOVE_DATA,
;  398          0779  4                   .COUNT, ODS1_HEADER + .ATR_TABLE [ .CODE, DISP ], .P, .ABD, .I, .ACCESS_MODE)
;  399          0780  3          ELSE IF .ATR_TABLE [ .CODE, ATR ] EQL ATR_ACMODE
;  400        P 0781  3          THEN KERNEL_CALL(MOVE_DATA,
;  401          0782  4                   .COUNT, DATA + .ATR_TABLE [ .CODE, DISP ], .P, .ABD, .I, .IO_PACKET[IRP$V_MODE])
;  402        P 0783  3          ELSE KERNEL_CALL(MOVE_DATA,
;  403          0784  3                   .COUNT, DATA + .ATR_TABLE [ .CODE, DISP ], .P, .ABD, .I, .ACCESS_MODE);
;  404          0785  2          END;
;  405          0786  2
;  406          0787  1      END;


                                                        .TITLE  RDATTR
                                                        .IDENT  \V04-000\

                                                        .PSECT  $DATA$,NOEXE,2

                                                00000 DATA::    .BLKB   512
                                                00200 ODS1_HEADER::
                                                        .BLKB   512
00  0E  20  00  00  0C  04  00  00  0A  03  00  00  08  05  00400 ATR_TABLE::
                                                        .BYTE   5, 8, 0, 0, 3, 10, 0, 0, 4, 12, 0, 0, 32, -
54  07  00  00  36  02  00  00  34  04  00  00  2E  05  00  0040F                   14, 0, 0, 15, 46, 0, 0, 4, 52, 0, 0, 2, -
50  00  02  00  02  00  00  00  00  00  01  00  20  00  00  0041E                   54, 0, 0, 7, 84, 0, 0, 32, 0, 1, 0, 0, 0, -
00  05  00  04  00  04  00  0E  00  00  38  23  00  03  00  0042D                   0, 0, 2, 0, 2, 0, 80, 0, 3, 0, 35, 56, 0, -
07  00  08  00  06  00  08  00  07  00  08  00  09  00  56  0043C                   0, 14, 0, 4, 0, 4, 0, 5, 0, 86, 0, 9, 0, -
00  02  00  00  38  02  00  08  00  04  00  06  00  08  00  0044B                   8, 0, 7, 0, 8, 0, 6, 0, 8, 0, 7, 0, 8, 0, -
02  00  08  00  04  00  06  00  08  00  06  00  01  00  06  0045A                   6, 0, 4, 0, 8, 0, 2, 56, 0, 0, 2, 0, 6, -
00  0A  00  01  00  06  00  02  00  04  00  06  00  04  00  00469                   0, 1, 0, 6, 0, 8, 0, 6, 0, 4, 0, 8, 0, 2, -
06  00  00  02  06  00  00  02  06  00  00  02  06  00  00  00478                   0, 4, 0, 6, 0, 4, 0, 2, 0, 6, 0, 1, 0, -
00  04  00  06  00  00  02  06  00  00  02  06  00  00  02  00487                   10, 0, 0, 0, 6, 2, 0, 0, 6, 2, 0, 0, 6, -
04  00  06  00  04  02  06  00  00  02  06  00  00  00  06  00496                   2, 0, 0, 6, 2, 0, 0, 6, 2, 0, 0, 6, 2, 0, -
00  0C  00  01  02  06  00  00  00  06  00  04  00  06  00  004A5                   0, 6, 0, 4, 0, 6, 0, 0, 0, 6, 2, 0, 0, 6, -
            00  0B  00  02  00  06  00  14  00  06  00  00  004B4                   2, 4, 0, 6, 0, 4, 0, 6, 0, 4, 0, 6, 0, 0, -
                                                                                    0, 6, 2, 1, 0, 12, 0, 0, 0, 6, 0, 20, 0, -
                                                                                    6, 0, 2, 0, 11, 0

                                                        .PSECT  $LOCKEDD1$,NOEXE,2

                                                00000 CURRENT_ATTRIB:
                                                        .BLKB   4

                                                        .EXTRN  CURRENT_UCB, HDR1
                                                        .EXTRN  HDR4, IO_PACKET
                                                        .EXTRN  FORMAT_FT1HOD1, LIB$CVT_DTB
                                                        .EXTRN  SYS$CMKRNL

                                                        .PSECT  $CODE$,NOWRT,2
```

```
                                                07FC 00000          .ENTRY  READ_ATTRIBUTE, Save R2,R3,R4,R5,R6,R7,R8,- ; 0655
                                                                            R9,R10
                                             5E         08  C2 00002       SUBL2   #8, SP
                                             6E         01  8A 00005       BICB2   #1, VALID_HEADER                        ; 0713
                                             50   0000G  CF  D0 00008       MOVL    IO_PACKET, R0                           ; 0714
                                5A      0B   A0        02  EF 0000D       EXTZV   #0, #2, 11(R0), ACCESS_MODE
                                             6D   0000V  CF  9E 00013       MOVAB   HANDLER, (FP)                           ; 0716
                                             7E     32   A0  3C 00018       MOVZWL  50(R0), -(SP)                           ; 0720
                                                         6E  D7 0001C       DECL    (SP)
                                             59     04   AC  D0 0001E       MOVL    ABD, R9                                 ; 0723
                                             57         05  D0 00022       MOVL    #5, I
                                                    0156 30  30 00025       BSBW    26$
                                        0000'   CF          57  D0 00028 1$:  MOVL    I, CURRENT_ATTRIB                       ; 0722
                                             50        6947  7E 0002D       MOVAQ   (R9)[I], R0                             ; 0723
                                             58          60  3C 00031       MOVZWL  (R0), P
                                             58          50  C0 00034       ADDL2   R0, P
                                                   02 A947  7F 00037       PUSHAQ  2(R9)[I]                                ; 0724
                                          0C    AE         9E  3C 0003B       MOVZWL  @(SP)+, COUNT
                                             56          68  9A 0003F       MOVZBL  (P), CODE                               ; 0725
                                                         56  D7 00042       DECL    CODE
                                             30          56  D1 00044       CMPL    CODE, #48                               ; 0733
                                                         02  1B 00047       BLEQU   2$
                                                         34  BF 00049       CHMU    #52
                                                0000'CF46    DF 0004B 2$:    PUSHAL  ATR_TABLE+3[CODE]                       ; 0739
                                             02         9E  91 00050       CMPB    @(SP)+, #2
                                                         0A  13 00053       BEQL    3$
                                                0000'CF46    DF 00055       PUSHAL  ATR_TABLE[CODE]                         ; 0740
                                             50         9E  9A 0005A       MOVZBL  @(SP)+, MAX_SIZE
                                                         05  11 0005D       BRB     4$
                                             50  017C   8F  3C 0005F 3$:    MOVZWL  #380, MAX_SIZE                          ; 0739
                                                         05  12 00064 4$:    BNEQ    5$                                      ; 0742
                                             50  0200   8F  3C 00066       MOVZWL  #512, MAX_SIZE
                                             50    0C   AE  D1 0006B 5$:    CMPL    COUNT, MAX_SIZE                         ; 0743
                                                         02  1B 0006F       BLEQU   6$
                                                         34  BF 00071       CHMU    #52
                                                0000'CF46    DF 00073 6$:    PUSHAL  ATR_TABLE+2[CODE]                       ; 0748
                                          0C                 00  8F 00078       CASEB   @(SP)+, #0, #12
                                003B           0034             002D       001A     0007C 7$:  .WORD   8$-7$,-
                                004A           003F             003B       009A     00084      .WORD   9$-7$,-
                                007B           006E             0064       0053     0008C      .WORD   10$-7$,-
                                                                           0082     00094      .WORD   11$-7$,-
                                                                                               .WORD   20$-7$,-
                                                                                               .WORD   11$-7$,-
                                                                                               .WORD   12$-7$,-
                                                                                               .WORD   13$-7$,-
                                                                                               .WORD   14$-7$,-
                                                                                               .WORD   15$-7$,-
                                                                                               .WORD   16$-7$,-
                                                                                               .WORD   17$-7$,-
                                                                                               .WORD   18$-7$
                                             7F          08  AE  E8 00096 8$:    BLBS    VALID_HEADER, 21$               ; 0750
                                                0000'    CF  9F 0009A       PUSHAB  ODS1_READER                             ; 0753
                                        0000G   CF          01  FB 0009E       CALLS   #1, FORMAT_F11HOD1
                                          08    AE          01  88 000A3       BISB2   #1, VALID_HEADER                    ; 0754
                                                            70  11 000A7       BRB     21$                                 ; 0750
                                        0000V   CF          00  FB 000A9 9$:    CALLS   #0, CALCULATE_STATISTICS            ; 0756
```

```
                                                69  11 000AE            BRB     21$
                        0000V   CF              00  FB 000B0 10$:       CALLS   #0, GET_BLOCK_SIZE                      ;0757
                                                62  11 000B5            BRB     21$
                                                34  BF 000B7 11$:       CHMU    #52                                     ;0760
                                                5E  11 000B9            BRB     21$                                     ;0748
        0C      AE              00              6E  00 2C 000BB 12$:       MOVC5   #0, (SP), #0, COUNT, DATA               ;0761
                                        0000'   CF      000C1
                                                53  11 000C4            BRB     21$
                                                56  DD 000C6 13$:       PUSHL   CODE                                    ;0762
                        0000V   CF              01  FB 000C8            CALLS   #1, CONVERT_DATE
                                                4A  11 000CD            BRB     21$
                                50      0000G   CF  D0 000CF 14$:       MOVL    CURRENT_UCB, R0                         ;0764
                                6E        1C    A0  D0 000D4            MOVL    28(R0), ORB
                        0000'   CF        00    BE  D0 000D8            MOVL    @ORB, DATA                              ;0765
                                                39  11 000DE            BRB     21$                                     ;0748
                                  0C    AE      9F 000E0 15$:       PUSHAB  COUNT                                   ;0767
                        0000V   CF              01  FB 000E3            CALLS   #1, ANSI_FILE_NAME
                                                2F  11 000E8            BRB     21$
                                50      0000G   CF  D0 000EA 16$:       MOVL    HDR1, R0                                ;0768
                        0000'   CF        35    A0  90 000EF            MOVB    53(R0), DATA
                                                22  11 000F5            BRB     21$
                        0000V   CF              00  FB 000F7 17$:       CALLS   #0, GET_BUFFER_OFFSET                   ;0769
                                                1B  11 000FC            BRB     21$
                                50      0000G   CF  D0 000FE 18$:       MOVL    IO_PACKET, R0                           ;0771
        50      0B      A0              02      00  EF 00103            EXTZV   #0, #2, 11(R0), R0
                                50        01    A8  91 00109            CMPB    1(P), R0
                                                04  1B 0010D            BLEQU   19$
                                50        01    A8  9A 0010F            MOVZBL  1(P), R0
                                5A              50  D0 00113 19$:       MOVL    R0, ACCESS_MODE
                                  0C    AE      D4 00116 20$:       CLRL    COUNT                                   ;0772
                                59        04    AC  D0 00119 21$:       MOVL    ABD, R9                                 ;0779
                                0000'CF   46    DF 0011D            PUSHAL  ATR_TABLE+2[CODE]                       ;0777
                                50              9E  9A 00122            MOVZBL  @(SP)+, R0
                                                19  12 00125            BNEQ    22$
                                0480    8F      BB 00127            PUSHR   #^M<R7,R10>                             ;0779
                                7E              58  7D 0012B            MOVQ    P, -(SP)
                                0000'CF   46    DF 0012E            PUSHAL  ATR_TABLE+1[CODE]
                                50              9E  9A 00133            MOVZBL  @(SP)+, R0
                                51      0000'   CF  9E 00136            MOVAB   ODS1_HEADER, R1
                                      6140      9F 0013B            PUSHAB  (R1)[R0]
                                                2A  11 0013E            BRB     25$
                                51      0000'   CF  9E 00140 22$:       MOVAB   DATA, R1                                ;0782
                                0000'CF   46    DF 00145            PUSHAL  ATR_TABLE+1[CODE]
                                52              9E  9A 0014A            MOVZBL  @(SP)+, R2
                                51              52  C0 0014D            ADDL2   R2, R1
                                0C              50  91 00150            CMPB    R0, #12                                 ;0780
                                                0D  12 00153            BNEQ    23$
                                50      0000G   CF  D0 00155            MOVL    IO_PACKET, R0                           ;0782
    7E      0B      A0              02              00  EF 0015A            EXTZV   #0, #2, 11(R0), -(SP)
                                                02  11 00160            BRB     24$
                                                5A  DD 00162 23$:       PUSHL   ACCESS_MODE                             ;0784
                                                57  DD 00164 24$:       PUSHL   I
                                0302    8F      BB 00166            PUSHR   #^M<R1,R8,R9>
                                  20    AE      DD 0016A 25$:       PUSHL   COUNT
                                                06  DD 0016D            PUSHL   #6
                                                5E  DD 0016F            PUSHL   SP
                                0000V   CF      9F 00171            PUSHAB  MOVE_DATA
```

```
                00000000G  9F          09  FB 00175          CALLS   #9, @#SYS$CMKRNL                    ;
                                       57  D6 0017C          INCL    I                                   ; 0720
                      04   AE          57  D1 0017E 26$:     CMPL    I, 4(SP)                            ;
                                       03  1A 00182          BGTRU   27$                                 ;
                                     FEA1  31 00184          BRW     1$                                  ;
                                           04 00187 27$:     RET                                         ; 0787

; Routine Size:  392 bytes,    Routine Base:  $CODE$ + 0000

;  407          0788  1
```

```
;  409     0789  1 ROUTINE MOVE_DATA (COUNT, SOURCE, DESTINATION, ABD, I, ACCESS_MODE) : COMMON_CALL NOVALUE =
;  410     0790  1
;  411     0791  1 !++
;  412     0792  1 !
;  413     0793  1 ! FUNCTIONAL DESCRIPTION:
;  414     0794  1 !       Write back of attribute enabled and attribute written into buffer packet
;  415     0795  1 !
;  416     0796  1 ! CALLING SEQUENCE:
;  417     0797  1 !       MOVE_DATA(ARG1,ARG2,ARG3,ARG4,ARG5,ARG6), called in kernel mode
;  418     0798  1 !
;  419     0799  1 ! INPUT PARAMETERS:
;  420     0800  1 !       ARG1 - number of characters to move
;  421     0801  1 !       ARG2 - source address
;  422     0802  1 !       ARG3 - destination address
;  423     0803  1 !       ARG4 - address of buffer descriptors
;  424     0804  1 !       ARG5 - number of current buffer descriptor
;  425     0805  1 !       ARG6 - access mode to set for buffer
;  426     0806  1 !
;  427     0807  1 ! IMPLICIT INPUTS:
;  428     0808  1 !       none
;  429     0809  1 !
;  430     0810  1 ! OUTPUT PARAMETERS:
;  431     0811  1 !       ARG3 - destination address
;  432     0812  1 !
;  433     0813  1 ! IMPLICIT OUTPUTS:
;  434     0814  1 !       none
;  435     0815  1 !
;  436     0816  1 ! ROUTINE VALUE:
;  437     0817  1 !       none
;  438     0818  1 !
;  439     0819  1 ! SIDE EFFECTS:
;  440     0820  1 !       none
;  441     0821  1 !
;  442     0822  1 !--
;  443     0823  1
;  444     0824  2     BEGIN
;  445     0825  2
;  446     0826  2     EXTERNAL REGISTER
;  447     0827  2         COMMON_REG;
;  448     0828  2
;  449     0829  2     MAP
;  450     0830  2         ABD     : REF BBLOCKVECTOR [, ABD$C_LENGTH];    ! address of buffer descriptors
;  451     0831  2
;  452     0832  2     ! set the buffered read bit in the I/O packet to indicate to IO_DONE that
;  453     0833  2     ! the attribute buffers are valid
;  454     0834  2     !
;  455     0835  2     IO_PACKET[IRP$V_FUNC] = 1;
;  456     0836  2
;  457     0837  2     ! restore access mode of user
;  458     0838  2     !
;  459     0839  2     (.DESTINATION)<0, 8> = .ACCESS_MODE;
;  460     0840  2
;  461     0841  2     ! return actual size of attribute to user
;  462     0842  2     !
;  463     0843  2     ABD[.I, ABD$W_COUNT] = .COUNT;
;  464     0844  2
;  465     0845  2     ! move attribute to buffer in system space
```

```
;  466          0846  2      !
;  467          0847  2      CH$MOVE(.COUNT, .SOURCE, .DESTINATION + 1);
;  468          0848  1      END;


                                                003C 00000 MOVE_DATA:
                                                           .WORD   Save R2,R3,R4,R5                    ; 0789
                                50      0000G   CF  D0 00002       MOVL    IO_PACKET, R0               ; 0835
                            2A  A0              02  88 00007       BISB2   #2, 42(R0)                  ;
                                51      0C      AC  D0 0000B       MOVL    DESTINATION, R1             ; 0839
                                61      18      AC  90 0000F       MOVB    ACCESS_MODE, (R1)           ;
                                50      14      AC  D0 00013       MOVL    I, R0                       ; 0843
                                50      10 BC40 7E  00017          MOVAQ   @ABD[R0], R0                ;
                            02  A0      04      AC  B0 0001C       MOVW    COUNT, 2(R0)                ;
                        01  A1      08  BC      04  AC 28 00021    MOVC3   COUNT, @SOURCE, 1(R1)       ; 0847
                                                   04 00028       RET                                 ; 0848

; Routine Size:  41 bytes,    Routine Base:  $CODE$ + 0188
```

```
;  470     0849  1 ROUTINE CALCULATE_STATISTICS : COMMON_CALL NOVALUE =
;  471     0850  1
;  472     0851  1 !++
;  473     0852  1 !
;  474     0853  1 ! FUNCTIONAL DESCRIPTION:
;  475     0854  1 !       calculate the statistics block
;  476     0855  1 !
;  477     0856  1 ! CALLING SEQUENCE:
;  478     0857  1 !       CALCULATE_STATISTICS
;  479     0858  1 !
;  480     0859  1 ! INPUT PARAMETERS:
;  481     0860  1 !       none
;  482     0861  1 !
;  483     0862  1 ! IMPLICIT INPUTS:
;  484     0863  1 !       DATA
;  485     0864  1 !
;  486     0865  1 ! OUTPUT PARAMETERS:
;  487     0866  1 !       none
;  488     0867  1 !
;  489     0868  1 ! IMPLICIT OUTPUTS:
;  490     0869  1 !       DATA get fill in
;  491     0870  1 !
;  492     0871  1 ! ROUTINE VALUE:
;  493     0872  1 !       none
;  494     0873  1 !
;  495     0874  1 ! SIDE EFFECTS:
;  496     0875  1 !       none
;  497     0876  1 !
;  498     0877  1 ! USER ERRORS:
;  499     0878  1 !       none
;  500     0879  1 !
;  501     0880  1 !--
;  502     0881  1
;  503     0882  2     BEGIN
;  504     0883  2
;  505     0884  2     EXTERNAL ROUTINE
;  506     0885  2         GTNEXT_VOL_READ : JSB,                  ! mount next volume for read
;  507     0886  2         READ_BLOCK      : COMMON_CALL,          ! read MT data block
;  508     0887  2         SPACE_TM        : COMMON_CALL;          ! space TM
;  509     0888  2
;  510     0889  2     EXTERNAL REGISTER
;  511     0890  2         COMMON_REG;
;  512     0891  2
;  513     0892  2     ! layout of statistics block
;  514     0893  2     !
;  515     0894  2     MACRO   LBN      = 0,0,32,0%;
;  516     0895  2     MACRO   BLOCKCNT = 4,0,32,0%;
;  517     0896  2     MACRO   ACCCNT   = 8,0,8,0%;
;  518     0897  2     MACRO   LCKCNT   = 9,0,8,0%;
;  519     0898  2
;  520     0899  2     EXTERNAL CURRENT_WCB    : REF BBLOCK;       ! address of current WCB
;  521     0900  2
;  522     0901  2     LOCAL    FUNCTION       : BLOCK [1];
;  523     0902  2
;  524     0903  2     FUNCTION = .IO_PACKET[IRP$W_FUNC];
;  525     0904  2
;  526     0905  2     !   If a window exists then the file has been accessed in which
```

```
;  527     0906  2      !   case do not bother to get block count.  Also do not return
;  528     0907  2      !   block count if file will be accessed.
;  529     0908  2      !
;  530     0909  2      IF .CURRENT_WCB NEQ 0  OR  .FUNCTION[IO$V_ACCESS]
;  531     0910  2      THEN
;  532     0911  3          BEGIN
;  533     0912  3          DATA[LBN]      = 0;
;  534     0913  3          DATA[BLOCKCNT] = 0;
;  535     0914  3          DATA[ACCCNT]   = 1;
;  536     0915  3          DATA[LCKCNT]   = 1;
;  537     0916  3          END
;  538     0917  3
;  539     0918  2      ELSE
;  540     0919  2          ! if partially created file then all statistics are zeroed
;  541     0920  2          !
;  542     0921  2          IF .CURRENT_VCB[VCB$V_PARTFILE]
;  543     0922  2          THEN
;  544     0923  2              CH$FILL(0, 10, DATA)
;  545     0924  2
;  546     0925  2          ELSE
;  547     0926  2              ! now if not accessed calculate block count
;  548     0927  2              !
;  549     0928  3              BEGIN
;  550     0929  3
;  551     0930  3              ! accumulative block total  This must be on the stack so it is
;  552     0931  3              ! saved when a block is done on multi-volume files
;  553     0932  3              !
;  554     0933  3              STACKLOCAL  ACCBLCNT;
;  555     0934  3
;  556     0935  3              LOCAL BLCNT;
;  557     0936  3
;  558     0937  3              ACCBLCNT = 0;
;  559     0938  3
;  560     0939  3              WHILE 1
;  561     0940  3              DO
;  562     0941  4                  BEGIN
;  563     0942  4
;  564     0943  4                  ! read trailers
;  565     0944  4                  !
;  566     0945  4                  SPACE_TM(2 - .CURRENT_VCB[VCB$B_TM]);
;  567     0946  4                  IF NOT READ_BLOCK(.HDR1, ANSI_LBLSZ)
;  568     0947  4                      THEN ERR_EXIT(SS$_TAPEPOSLOST);
;  569     0948  4                  LIB$CVT_DTB(EO1$S_BLOCKCNT, HDR1[EO1$T_BLOCKCNT], BLCNT);
;  570     0949  4                  ACCBLCNT = .ACCBLCNT + .BLCNT;
;  571     0950  4                  IF .HDR1[EO1$L_EO1LID] EQL 'EOF1' THEN EXITLOOP;
;  572     0951  4                  GTNEXT_VOL_READ();
;  573     0952  3                  END;
;  574     0953  3
;  575     0954  3              DATA[LBN]      = 0;
;  576     0955  3              DATA[BLOCKCNT] = ROT(.ACCBLCNT, 16);
;  577     0956  3              DATA[ACCCNT]   = 0;
;  578     0957  3              DATA[LCKCNT]   = 0;
;  579     0958  2              END;
;  580     0959  2
;  581     0960  1      END;                            ! end of routine
```

```
                                                        .EXTRN  GTNEXT_VOL_READ
                                                        .EXTRN  READ_BLOCK, SPACE_TM
                                                        .EXTRN  CURRENT_WCB

                                        07FC 00000 CALCULATE_STATISTICS:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10                ; 0849
                                    56      0000'   CF  9E 00002       MOVAB   DATA, R6                        ;
                                    5E              08  C2 00007       SUBL2   #8, SP                          ;
                                    50      0000G   CF  D0 0000A       MOVL    IO_PACKET, R0                   ; 0903
                                    50        20    A0  3C 0000F       MOVZWL  32(R0), FUNCTION                ;
                                            0000G   CF  D5 00013       TSTL    CURRENT_WCB                     ; 0909
                                                    04  12 00017       BNEQ    1$                              ;
                            09              50          06  E1 00019   BBC     #6, FUNCTION, 2$                ;
                                                    66  7C 0001D 1$:    CLRQ    DATA                            ; 0912
                                    08      A6  0101    8F  B0 0001F    MOVW    #257, DATA+8                    ; 0914
                                                        04  00025       RET                                     ; 0909
                                            07    0B    AB  E9 00026 2$:  BLBC    11(CURRENT_VCB), 3$             ; 0921
                        0A          00      6E          00  2C 0002A   MOVC5   #0, (SP), #0, #10, DATA           ; 0923
                                                    66      0002F       ;                                       ;
                                                        04  00030       RET                                     ;
                                              04    AE    D4  00031 3$:  CLRL    ACCBLCNT                        ; 0937
                                            7E      2E    AB  9A 00034 4$:  MOVZBL  46(CURRENT_VCB), -(SP)          ; 0945
                                    6E              02          6E  C3 00038   SUBL3   (SP), #2, (SP)            ;
                                            0000G   CF          01  FB 0003C   CALLS   #1, SPACE_TM              ;
                                            7E        50    8F  9A 00041    MOVZBL  #80, -(SP)                      ; 0946
                                                    0000G   CF  DD 00045    PUSHL   HDR1                            ;
                                            0000G   CF          02  FB 00049   CALLS   #2, READ_BLOCK           ;
                                                    04          50  E8 0004E   BLBS    R0, 5$                  ;
                                                        0224    8F  BF 00051    CHMU    #548                    ; 0947
                                                                5E  DD 00055 5$:  PUSHL   SP                      ; 0948
                                                7E      0000G   CF  36  C1 00057   ADDL3   #54, HDR1, -(SP)      ;
                                                                06  DD 0005D   PUSHL   #6                       ;
                                                    00000000G   9F  03  FB 0005F   CALLS   #3, @#LIB$CVT_DTB    ;
                                                            04  AE  6E  C0 00066   ADDL2   BLCNT, ACCBLCNT       ; 0949
                                                    31464F45    8F  D1  0006A   CMPL    @HDR1, #826691397       ; 0950
                                                                05  13  00073   BEQL    6$                      ;
                                                                0000G 30 00075   BSBW    GTNEXT_VOL_READ         ; 0951
                                                                BA  11  00078   BRB     4$                      ; 0939
                                                                66  D4  0007A 6$:  CLRL    DATA                    ; 0954
                                                04  A6      04  AE  10  9C 0007C   ROTL    #16, ACCBLCNT, DATA+4   ; 0955
                                                                08  A6  B4 00082   CLRW    DATA+8                  ; 0956
                                                                    04  00085   RET                             ; 0960

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

```
;  583          0961  1 ROUTINE GET_BLOCK_SIZE : NOVALUE =
;  584          0962  1
;  585          0963  1 !++
;  586          0964  1 !
;  587          0965  1 ! FUNCTIONAL DESCRIPTION:
;  588          0966  1 !       get block size from HDR2 if availible else return 512
;  589          0967  1 !
;  590          0968  1 ! CALLING SEQUENCE:
;  591          0969  1 !       GET_BLOCK_SIZE
;  592          0970  1 !
;  593          0971  1 ! INPUT PARAMETERS:
;  594          0972  1 !       none
;  595          0973  1 !
;  596          0974  1 ! IMPLICIT INPUTS:
;  597          0975  1 !       DATA
;  598          0976  1 !
;  599          0977  1 ! OUTPUT PARAMETERS:
;  600          0978  1 !       none
;  601          0979  1 !
;  602          0980  1 ! IMPLICIT OUTPUTS:
;  603          0981  1 !       DATA get fill in
;  604          0982  1 !
;  605          0983  1 ! ROUTINE VALUE:
;  606          0984  1 !       none
;  607          0985  1 !
;  608          0986  1 ! SIDE EFFECTS:
;  609          0987  1 !       none
;  610          0988  1 !
;  611          0989  1 ! USER ERRORS:
;  612          0990  1 !       none
;  613          0991  1 !
;  614          0992  1 !--
;  615          0993  1
;  616          0994  2     BEGIN
;  617          0995  2
;  618          0996  2     EXTERNAL HDR2 : REF BBLOCK;         ! address of HDR2(EOF2) label
;  619          0997  2
;  620          0998  2     LOCAL RESULT;
;  621          0999  2
;  622          1000  2     IF NOT LIB$CVT_DTB(HD2$S_BLOCKLEN, HDR2[HD2$T_BLOCKLEN], RESULT)
;  623          1001  2     THEN RESULT = 512;
;  624          1002  2
;  625          1003  2     DATA<0, 16> = .RESULT<0, 16>;
;  626          1004  1     END;


                                                        .EXTRN  HDR2

                                        0000 00000 GET_BLOCK_SIZE:
                                                        .WORD   Save nothing                    ; 0961
                            5E          04   C2 00002           SUBL2   #4, SP                  ; 1000
                                        5E   DD 00005           PUSHL   SP
                    7E      0000G   CF  05   C1 00007           ADDL3   #5, HDR2, -(SP)
                                        05   DD 0000D           PUSHL   #5
                00000000G   9F          03   FB 0000F           CALLS   #3, @#LIB$CVT_DTB
                            05          50   E8 00016           BLBS    R0, 1$                  ;
```

```
                     6E      0200    8F  3C 00019          MOVZWL  #512, RESULT                    ; 1001
                0000' CF             6E  B0 0001E 1$:      MOVW    RESULT, DATA                   ; 1003
                                         04 00023          RET                                    ; 1004

; Routine Size:  36 bytes,    Routine Base:  $CODE$ + 0237


;  627          1005  1
;  628          1006  1 ROUTINE GET_BUFFER_OFFSET : NOVALUE =
;  629          1007  1
;  630          1008  1 !++
;  631          1009  1 !
;  632          1010  1 ! FUNCTIONAL DESCRIPTION:
;  633          1011  1 !       Get buffer offset length from HDR2.
;  634          1012  1 !
;  635          1013  1 ! CALLING SEQUENCE:
;  636          1014  1 !       GET_BUFFER_OFFSET
;  637          1015  1 !
;  638          1016  1 ! INPUT PARAMETERS:
;  639          1017  1 !       none
;  640          1018  1 !
;  641          1019  1 ! IMPLICIT INPUTS:
;  642          1020  1 !       DATA
;  643          1021  1 !
;  644          1022  1 ! OUTPUT PARAMETERS:
;  645          1023  1 !       none
;  646          1024  1 !
;  647          1025  1 ! IMPLICIT OUTPUTS:
;  648          1026  1 !       The buffer offset length gets returned in the DATA field
;  649          1027  1 !
;  650          1028  1 ! ROUTINE VALUE:
;  651          1029  1 !       none
;  652          1030  1 !
;  653          1031  1 ! SIDE EFFECTS:
;  654          1032  1 !       none
;  655          1033  1 !
;  656          1034  1 ! USER ERRORS:
;  657          1035  1 !       none
;  658          1036  1 !
;  659          1037  1 !--
;  660          1038  1
;  661          1039  2     BEGIN
;  662          1040  2
;  663          1041  2     EXTERNAL HDR2 : REF BBLOCK;          ! address of HDR2(EOF2) label
;  664          1042  2
;  665          1043  2     LOCAL RESULT;
;  666          1044  2
;  667          1045  2     LIB$CVT_DTB(HD2$S_BUFOFF, HDR2[HD2$T_BUFOFF], RESULT);
;  668          1046  2     DATA<0, 16> = .RESULT<0, 16>;
;  669          1047  1     END;


                                0000 00000 GET_BUFFER_OFFSET:
                                                .WORD   Save nothing                    ; 1006
```

```
                          5E        04  C2 00002        SUBL2   #4, SP                                  ;
                          5E        DD  00005        PUSHL   SP                                       ; 1045
                7E      0000G  CF  32  C1 00007        ADDL3   #50, HDR2, -(SP)                         ;
                          02        DD  0000D        PUSHL   #2                                       ;
             00000000G    9F  03  FB  0000F        CALLS   #3, @#LIB$CVT_DTB                        ;
                0000'     CF  6E  B0  00016        MOVW    RESULT, DATA                             ; 1046
                          04        0001B        RET                                              ; 1047

; Routine Size:  28 bytes,    Routine Base:  $CODE$ + 025B

;  670          1048  1
```

```
;  672          1049  1 ROUTINE CONVERT_DATE ( CODE ) : NOVALUE =
;  673          1050  1
;  674          1051  1 !++
;  675          1052  1 !
;  676          1053  1 ! FUNCTIONAL DESCRIPTION:
;  677          1054  1 !       convert the date from ANSI Julian to 64 bit binary
;  678          1055  1 !
;  679          1056  1 ! CALLING SEQUENCE:
;  680          1057  1 !       CONVERT_DATE
;  681          1058  1 !
;  682          1059  1 ! INPUT PARAMETERS:
;  683          1060  1 !       CODE - the attribute code
;  684          1061  1 !
;  685          1062  1 ! IMPLICIT INPUTS:
;  686          1063  1 !       DATA
;  687          1064  1 !
;  688          1065  1 ! OUTPUT PARAMETERS:
;  689          1066  1 !       none
;  690          1067  1 !
;  691          1068  1 ! IMPLICIT OUTPUTS:
;  692          1069  1 !       DATA get fill in
;  693          1070  1 !
;  694          1071  1 ! ROUTINE VALUE:
;  695          1072  1 !       none
;  696          1073  1 !
;  697          1074  1 ! SIDE EFFECTS:
;  698          1075  1 !       none
;  699          1076  1 !
;  700          1077  1 ! USER ERRORS:
;  701          1078  1 !       none
;  702          1079  1 !
;  703          1080  1 !--
;  704          1081  1
;  705          1082  1
;  706          1083  2     BEGIN
;  707          1084  2
;  708          1085  2     LOCAL
;  709          1086  2         ADDR    : REF BBLOCK;
;  710          1087  2
;  711          1088  2     EXTERNAL ROUTINE
;  712          1089  2         CONVDATE_J2R,                                   ! convert Julian to reg date
;  713          1090  2         SYS$BINTIM : ADDRESSING_MODE (ABSOLUTE);! ASCII to 64 bit time
;  714          1091  2
;  715          1092  2     IF (.CODE + 1) EQLU ATR$C_CREDATE
;  716          1093  2     THEN ADDR = HDR1[HD1$T_CREATEDT]
;  717          1094  2     ELSE ADDR = HDR1[HD1$T_EXPIREDT];
;  718          1095  2
;  719          1096  2     IF CONVDATE_J2R(DATA + 8, .ADDR)
;  720          1097  2     THEN
;  721          1098  3         BEGIN
;  722          1099  3
;  723          1100  3         ! append a zero time to the date string
;  724          1101  3         ! set up a descriptor to the date string
;  725          1102  3         ! and convert to 64 bit number
;  726          1103  3         !
;  727          1104  3         BIND NO_TIME = UPLIT BYTE ( ' 00:00:00.00' );
;  728          1105  3
```

```
;  729          1106  3          CH$MOVE ( 12, NO_TIME, DATA + 19 );
;  730          1107  3          (DATA + 32) = 23;
;  731          1108  3          (DATA + 36) = DATA + 8;
;  732          1109  3          SYS$BINTIM(DATA + 32, DATA);
;  733          1110  3          END
;  734          1111  3
;  735          1112  2      ELSE
;  736          1113  2
;  737          1114  2          ! date was bad or no date, return binary zero
;  738          1115  2          !
;  739          1116  2          CH$FILL ( 0, 8, DATA );
;  740          1117  1      END;                                    ! end of routine


                30 30 2E 30 30 3A 30 30 3A 30 30 20  00277 P.AAA:  .ASCII  \ 00:00:00.00\                    ;

                                                             NO_TIME=          P.AAA
                                                                    .EXTRN  CONVDATE_J2R, SYS$BINTIM

                                                      007C 00000 CONVERT_DATE:
                                                                    .WORD   Save R2,R3,R4,R5,R6               ; 1049
                                          56     0000'  CF  9E 00002        MOVAB   DATA+8, R6
                                50        04     AC     01  C1 00007        ADDL3   #1, CODE, R0               ; 1092
                                               11     50  D1 0000C        CMPL    R0, #17
                                                      08  12 0000F        BNEQ    1$
                                50      0000G  CF     29  C1 00011        ADDL3   #41, HDR1, ADDR            ; 1093
                                                      06  11 00017        BRB     2$
                                50      0000G  CF     2F  C1 00019 1$:     ADDL3   #47, HDR1, ADDR            ; 1094
                                                      50  DD 0001F 2$:     PUSHL   ADDR                       ; 1096
                                                      56  DD 00021        PUSHL   R6
                                        0000G  CF     02  FB 00023        CALLS   #2, CONVDATE_J2R
                                               1C     50  E9 00028        BLBC    R0, 3$
                          0B    A6          C5     AF     0C  28 0002B        MOVC3   #12, NO_TIME, DATA+19      ; 1106
                                            18     A6     17  D0 00031        MOVL    #23, DATA+32               ; 1107
                                            1C     A6     66  9E 00035        MOVAB   DATA+8, DATA+36            ; 1108
                                                      F8  A6  9F 00039        PUSHAB  DATA                       ; 1109
                                                      18  A6  9F 0003C        PUSHAB  DATA+32
                                   00000000G  9F         02  FB 0003F        CALLS   #2, a#SYS$BINTIM
                                                      04  00046        RET                                ; 1096
                08                  00            6E     00  2C 00047 3$:     MOVC5   #0, (SP), #0, #8, DATA     ; 1116
                                                      F8  A6     0004C
                                                      04  0004E        RET                                ; 1117

; Routine Size:  79 bytes,    Routine Base:  $CODE$ + 0283
```

```
;  742          1118  1 ROUTINE ANSI_FILE_NAME ( COUNT ) : COMMON_CALL NOVALUE =
;  743          1119  1
;  744          1120  1 !++
;  745          1121  1 !
;  746          1122  1 ! FUNCTIONAL DESCRIPTION:
;  747          1123  1 !       Return the the ANSI file name from HDR1 and the HDR4 labels
;  748          1124  1 !       in the ASCII name attribute
;  749          1125  1 !
;  750          1126  1 ! CALLING SEQUENCE:
;  751          1127  1 !       ANSI_FILE_NAME
;  752          1128  1 !
;  753          1129  1 ! INPUT PARAMETERS:
;  754          1130  1 !       count - the size of the users buffer
;  755          1131  1 !
;  756          1132  1 ! IMPLICIT INPUTS:
;  757          1133  1 !       DATA
;  758          1134  1 !
;  759          1135  1 ! OUTPUT PARAMETERS:
;  760          1136  1 !       count - the size of the users buffer minus trailing spaces
;  761          1137  1 !
;  762          1138  1 ! IMPLICIT OUTPUTS:
;  763          1139  1 !       the name gets filled into DATA
;  764          1140  1 !
;  765          1141  1 ! ROUTINE VALUE:
;  766          1142  1 !       none
;  767          1143  1 !
;  768          1144  1 ! SIDE EFFECTS:
;  769          1145  1 !       none
;  770          1146  1 !
;  771          1147  1 ! USER ERRORS:
;  772          1148  1 !       SS$_BADATTRIB - if the buffer is to small
;  773          1149  1 !
;  774          1150  1 !--
;  775          1151  1
;  776          1152  2     BEGIN
;  777          1153  2
;  778          1154  2     EXTERNAL REGISTER
;  779          1155  2         COMMON_REG;
;  780          1156  2
;  781          1157  2     MAP
;  782          1158  2         COUNT           : REF VECTOR [ 1, LONG ];       ! pointer to long word
;  783          1159  2
;  784          1160  2     LOCAL
;  785          1161  2         FILE_ID         : VECTOR [ FILE_SPEC_MAX, BYTE ], ! vector of characters
;  786          1162  2         MVL             : REF BBLOCK,                   ! MVL address
;  787          1163  2         STRIPED_SIZE    : LONG INITIAL ( 0 );           ! file id size minus
;  788          1164  2                                                         ! trailing spaces
;  789          1165  2
;  790          1166  2     ! Space fill the temporary file identifier field
;  791          1167  2
;  792          1168  2     CH$FILL (' ',FILE_SPEC_MAX,FILE_ID);
;  793          1169  2
;  794          1170  2     ! Copy the file identifier from the HDR1 and HDR4 labels into a local field
;  795          1171  2     ! Pleae note that the size of the FILEID extension stored in the HDR4 field
;  796          1172  2     ! is stored in different locations depending on the ANSI version stored
;  797          1173  2     ! in the VOL1 label.
;  798          1174  2
```

```
;  799          1175  2      ! First copy the HDR1 file id
;  800          1176  2
;  801          1177  2      CH$MOVE(HD1$S_FILEID, HDR1[HD1$T_FILEID], FILE_ID[0]);
;  802          1178  2
;  803          1179  2      ! The get the HDR4 fiel id extension
;  804          1180  2
;  805          1181  2      MVL  = .CURRENT_VCB[VCB$L_MVL];
;  806          1182  2      IF .MVL[MVL$B_STDVER] GTR 3
;  807          1183  2          THEN
;  808          1184  2          BEGIN
;  809          1185  2              IF .HDR4[HD4$B_FILEID_EXT_SIZE] GTRU 0
;  810          1186  2                  THEN
;  811          1187  2                      CH$MOVE(.HDR4[HD4$B_FILEID_EXT_SIZE], HDR4[HD4$T_FILEID_EXT],
;  812          1188  2                                                  FILE_ID[HD1$S_FILEID]);
;  813          1189  2          END
;  814          1190  2          ELSE
;  815          1191  2          BEGIN
;  816          1192  2              LOCAL SIZE;
;  817          1193  2              IF NOT LIB$CVT_DTB(HD4$S_FILEID_EXT_V3,
;  818          1194  2                                 HDR4[HD4$T_FILEID_EXT_V3], SIZE)
;  819          1195  2                  THEN SIZE = 0
;  820          1196  2                  ELSE
;  821          1197  2                      CH$MOVE(.SIZE, HDR4[HD4$T_FILEID_EXT],
;  822          1198  2                              FILE_ID[HD1$S_FILEID]);
;  823          1199  2          END;
;  824          1200  2
;  825          1201  2      ! find the size of the File Id minus trailing spaces
;  826          1202  2
;  827          1203  2      DECR I FROM (FILE_SPEC_MAX - 1) TO 0 DO
;  828          1204  2          IF .FILE_ID [ .I ] NEQ ' '
;  829          1205  2          THEN
;  830          1206  3              BEGIN
;  831          1207  3              STRIPED_SIZE = .I + 1;
;  832          1208  3              EXITLOOP
;  833          1209  2              END;
;  834          1210  2
;  835          1211  2      ! if the file id is to large for buffer return error
;  836          1212  2
;  837          1213  2      IF .STRIPED_SIZE GTRU .COUNT [ 0 ] THEN ERR_EXIT ( SS$_BADATTRIB );
;  838          1214  2
;  839          1215  2      ! copy the id to the return buffer padd with spaces
;  840          1216  2
;  841          1217  2      CH$COPY ( FILE_SPEC_MAX, FILE_ID[0], ' ', .COUNT [ 0 ], DATA );
;  842          1218  2
;  843          1219  2      ! return the actual size of the file id
;  844          1220  2
;  845          1221  2      COUNT = .STRIPED_SIZE;
;  846          1222  2
;  847          1223  1      END;


                                007C 00000 ANSI_FILE_NAME:
                                           .WORD   Save R2,R3,R4,R5,R6                     ; 1118
                     5E        AC    AE  9E 00002         MOVAB   -84(SP), SP             ;
```

```
                                                56  D4 00006          CLRL    STRIPED_SIZE                                    ; 1152
    004F    8F          20          6E                  00  2C 00008  MOVC5   #0, (SP), #32, #79, FILE_ID                    ; 1168
                                                    04  AE     0000F
                                                50  0000G CF  D0 00011  MOVL    HDR1, R0                                        ; 1177
                04  AE          04  A0                  11  28 00016  MOVC3   #17, 4(R0), FILE_ID
                                                51  34    AB  D0 0001C  MOVL    52(CURRENT_VCB), MVL                            ; 1181
                                                50  0000G CF  D0 00020  MOVL    HDR4, R0                                        ; 1185
                                                03  22    A1  91 00025  CMPB    34(MVL), #3                                     ; 1182
                                                            11  1B 00029  BLEQU   1$
                                                    04  A0  95 0002B  TSTB    4(R0)                                           ; 1185
                                                            2C  13 0002E  BEQL    3$
                                                51  04  A0  9A 00030  MOVZBL  4(R0), R1                                       ; 1187
                15  AE          05  A0                  51  28 00034  MOVC3   R1, 5(R0), FILE_ID+17                          ; 1188
                                                            20  11 0003A  BRB     3$                                           ; 1182
                                                            5E  DD 0003C 1$:  PUSHL   SP                                      ; 1194
                                                    43  A0  9F 0003E  PUSHAB  67(R0)
                                                            02  DD 00041  PUSHL   #2
                        00000000G   9F                      03  FB 00043  CALLS   #3, @#LIB$CVT_DTB
                                                04                  50  E8 0004A  BLBS    R0, 2$
                                                            6E  D4 0004D  CLRL    SIZE                                        ; 1195
                                                            0B  11 0004F  BRB     3$
                                                50  0000G CF  D0 00051 2$:  MOVL    HDR4, R0                                    ; 1197
                15  AE          05  A0                  6E  28 00056  MOVC3   SIZE, 5(R0), FILE_ID+17                       ; 1198
                                                50  4E    8F  9A 0005C 3$:  MOVZBL  #78, I                                    ; 1203
                                                20  04  AE40  91 00060 4$:  CMPB    FILE_ID[I], #32                             ; 1204
                                                            06  13 00065  BEQL    5$
                                                56  01    A0  9E 00067  MOVAB   1(R0), STRIPED_SIZE                         ; 1207
                                                            03  11 0006B  BRB     6$                                          ; 1206
                                                F0                  50  F4 0006D 5$:  SOBGEQ  I, 4$                          ; 1204
                                                    04  BC          56  D1 00070 6$:  CMPL    STRIPED_SIZE, @COUNT            ; 1213
                                                            02  1B 00074  BLEQU   7$
                                                            34  BF 00076  CHMU    #52
    04  BC              20          04  AE      004F    8F  2C 00078 7$:  MOVC5   #79, FILE_ID, #32, @COUNT, DATA             ; 1217
                                                    0000'   CF      00081
                                                    04  AC          56  D0 00084  MOVL    STRIPED_SIZE, COUNT                 ; 1221
                                                                    04 00088  RET                                             ; 1223

; Routine Size:  137 bytes,    Routine Base:  $CODE$ + 02D2
```

```
;  849          1224  1 ROUTINE HANDLER : COMMON_CALL NOVALUE =
;  850          1225  1
;  851          1226  1 !++
;  852          1227  1 !
;  853          1228  1 ! FUNCTIONAL DESCRIPTION:
;  854          1229  1 !       This routine does clean-up on attribute error
;  855          1230  1 !
;  856          1231  1 ! CALLING SEQUENCE:
;  857          1232  1 !       HANDLER()
;  858          1233  1 !
;  859          1234  1 ! INPUT PARAMETERS:
;  860          1235  1 !       none
;  861          1236  1 !
;  862          1237  1 ! IMPLICIT INPUTS:
;  863          1238  1 !       CURRENT_AITRIB - address of descriptor number for current attribute
;  864          1239  1 !
;  865          1240  1 ! OUTPUT PARAMETERS:
;  866          1241  1 !       none
;  867          1242  1 !
;  868          1243  1 ! IMPLICIT OUTPUTS:
;  869          1244  1 !       none
;  870          1245  1 !
;  871          1246  1 ! ROUTINE VALUE:
;  872          1247  1 !       none
;  873          1248  1 !
;  874          1249  1 ! SIDE EFFECTS:
;  875          1250  1 !       none
;  876          1251  1 !
;  877          1252  1 ! USER ERRORS:
;  878          1253  1 !       none
;  879          1254  1 !
;  880          1255  1 !--
;  881          1256  1
;  882          1257  2     BEGIN
;  883          1258  2
;  884          1259  2     EXTERNAL REGISTER
;  885          1260  2         COMMON_REG;
;  886          1261  2
;  887          1262  2     KERNEL_CALL(SET_BCNT, .CURRENT_ATTRIB);
;  888          1263  1     END;


                                        0000 00000 HANDLER:.WORD    Save nothing                    ; 1224
                              0000'  CF   DD 00002          PUSHL   CURRENT_ATTRIB                  ; 1262
                                     01   DD 00006          PUSHL   #1
                                     5E   DD 00008          PUSHL   SP
                              0000V  CF   9F 0000A          PUSHAB  SET_BCNT
                              00000000G  9F                 04   FB 0000E          CALLS   #4, @#SYS$CMKRNL
                                                            04 00015          RET                     ; 1263

; Routine Size:  22 bytes,    Routine Base:  $CODE$ + 035B
```

```
;  890          1264  1 ROUTINE SET_BCNT (NUMBER) : COMMON_CALL NOVALUE =
;  891          1265  1
;  892          1266  1 !++
;  893          1267  1 !
;  894          1268  1 ! FUNCTIONAL DESCRIPTION:
;  895          1269  1 !       This routine sets the buffer count to the number of valid buffer descriptors
;  896          1270  1 !
;  897          1271  1 ! CALLING SEQUENCE:
;  898          1272  1 !       SET_BCNT(ARG1), called in kernel mode
;  899          1273  1 !
;  900          1274  1 ! INPUT PARAMETERS:
;  901          1275  1 !       The number of valid buffer descriptor
;  902          1276  1 !
;  903          1277  1 ! IMPLICIT INPUTS:
;  904          1278  1 !       IO_PACKET - address of current IO request packet
;  905          1279  1 !
;  906          1280  1 ! OUTPUT PARAMETERS:
;  907          1281  1 !       none
;  908          1282  1 !
;  909          1283  1 ! IMPLICIT OUTPUTS:
;  910          1284  1 !       none
;  911          1285  1 !
;  912          1286  1 ! ROUTINE VALUE:
;  913          1287  1 !       none
;  914          1288  1 !
;  915          1289  1 ! SIDE EFFECTS:
;  916          1290  1 !       none
;  917          1291  1 !
;  918          1292  1 ! USER ERRORS:
;  919          1293  1 !       none
;  920          1294  1 !
;  921          1295  1 !--
;  922          1296  1
;  923          1297  2     BEGIN
;  924          1298  2
;  925          1299  2     EXTERNAL REGISTER
;  926          1300  2         COMMON_REG;
;  927          1301  2
;  928          1302  2     IO_PACKET[IRP$W_BCNT] = .NUMBER;
;  929          1303  1     END;


                                        0000 00000 SET_BCNT:
                                                   .WORD   Save nothing                    ; 1264
                    50      0000G   CF  D0 00002           MOVL    IO_PACKET, R0           ; 1302
                32  A0        04    AC  B0 00007           MOVW    NUMBER, 50(R0)          ;
                                        04 0000C           RET                             ; 1303

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 0371

;  930          1304  1
```

```
;  932          1305  1 !++++
;  933          1306  1 GLOBAL ROUTINE COMPLETE_USRLBL (AST_BLOCK, NUMBER) : COMMON_CALL NOVALUE =
;  934          1307  1 !
;  935          1308  1 !++
;  936          1309  1 !
;  937          1310  1 ! FUNCTIONAL DESCRIPTION:
;  938          1311  1 !       This routine releases the current AST block, stores the users next AST
;  939          1312  1 !       control block and completes the current IO request
;  940          1313  1 !
;  941          1314  1 ! CALLING SEQUENCE:
;  942          1315  1 !       COMPLETE_USRLBL(ARG1,ARG2), called in kernel mode
;  943          1316  1 !
;  944          1317  1 ! INPUT PARAMETERS:
;  945          1318  1 !       ARG1 - address of user supplied AST control block
;  946          1319  1 !       ARG2 - number of descriptor for current attribute
;  947          1320  1 !
;  948          1321  1 ! IMPLICIT INPUTS·
;  949          1322  1 !       none
;  950          1323  1 !
;  951          1324  1 ! OUTPUT PARAMETERS:
;  952          1325  1 !       none
;  953          1326  1 !
;  954          1327  1 ! IMPLICIT OUTPUTS:
;  955          1328  1 !       none
;  956          1329  1 !
;  957          1330  1 ! ROUTINE VALUE:
;  958          1331  1 !       none
;  959          1332  1 !
;  960          1333  1 ! SIDE EFFECTS:
;  961          1334  1 !       none
;  962          1335  1 !
;  963          1336  1 ! USER ERRORS:
;  964          1337  1 !       none
;  965          1338  1 !
;  966          1339  1 !--
;  967          1340  1
;  968          1341  1     BEGIN
;  969          1342  1
;  970          1343  1     EXTERNAL REGISTER
;  971          1344  1         COMMON_REG;
;  972          1345  1
;  973          1346  1     LOCAL
;  974          1347  1         PCB     : REF BBLOCK;
;  975          1348  1
;  976          1349  1     IF .CURRENT_VCB[VCB$L_USRLBLAST] NEQ 0
;  977          1350  1     THEN
;  978          1351  1         ! if one currently recorded
;  979          1352  1         !
;  980          1353  1         BEGIN
;  981          1354  1         DEALLOCATE(.CURRENT_VCB[VCB$L_USRLBLAST]);      ! deallocate memory
;  982          1355  1
;  983          1356  1         ! inc user AST quota
;  984          1357  1         !
;  985          1358  1         PCB = .SCH$GL_PCBVEC[.(IO_PACKET[IRP$L_PID])<0, 16>];
;  986          1359  1         PCB[PCB$W_ASTCNT] = .PCB[PCB$W_ASTCNT] + 1;
;  987          1360  1         END;
;  988          1361  1 !
```

```
;   989          1362  1 !     CURRENT_VCB[VCB$L_USRLBLAST] = .AST_BLOCK;
;   990          1363  1 !
;   991          1364  1 !     IF .CURRENT_VCB[VCB$V_WAIUSRLBL]
;   992          1365  1 !     THEN
;   993          1366  1 !         BEGIN
;   994          1367  1 !         IO_DONE(.IO_PACKET);
;   995          1368  1 !         IO_PACKET = 0;                  ! note packet already returned to user
;   996          1369  1 !         END;
;   997          1370  1 !
;   998          1371  1 !     END;
;   999          1372  1 !----
```

```
;  1001          1373  1 !+          LOCAL
;  1002          1374  1 !                 AST_BLOCK : REF BBLOCK,
;  1003          1375  1 !                 MODE,
;  1004          1376  1 !                 LENGTH;
;  1005          1377  1 !
;  1006          1378  1 !             AST_BLOCK = .(.P+1);            ! pickup attribute following attr code
;  1007          1379  1 !
;  1008          1380  1 !             ! if AST address given then it must be a valid AST type control block
;  1009          1381  1 !             !
;  1010          1382  1 !             BEGIN
;  1011          1383  1 !             BUILTIN
;  1012          1384  1 !                 PROBER;
;  1013          1385  1 !             MODE = 0;
;  1014          1386  1 !             LENGTH = 4;
;  1015          1387  1 !             IF  .AST_BLOCK NEQ 0                                ! check for no AST block given
;  1016          1388  1 !                 AND
;  1017          1389  1 !                 (NOT PROBER(MODE,LENGTH,.AST_BLOCK)             ! check valid address
;  1018          1390  1 !                     OR
;  1019          1391  1 !                 .AST_BLOCK[ACB$B_TYPE] NEQ DYN$C_ACB)           ! check AST block type
;  1020          1392  1 !             THEN
;  1021          1393  1 !                 ERR_EXIT(SS$_ILLLBLAST);
;  1022          1394  1 !             END;
;  1023          1395  1 !             KERNEL_CALL(COMPLETE_USRLBL,.AST_BLOCK,.I);
;  1024          1396  1 !             IF .CURRENT_VCB[VCB$V_WAIUSRLBL]
;  1025          1397  1 !             THEN
;  1026          1398  1 !                 UNBLOCK(.CURRENT_VCB);
;  1027          1399  1 !-
```

```
;1029     1400  1 !*          LOCAL
;1030     1401  1 !                 STRATCH : REF BBLOCK;
;1031     1402  1 !
;1032     1403  1 !           ! if not reading or not waiting for user label read or not EOF then error
;1033     1404  1 !           !
;1034     1405  1 !           IF  NOT .CURRENT_WCB[WCB$V_READ]
;1035     1406  1 !               OR
;1036     1407  1 !               (NOT .CURRENT_VCB[VCB$V_WAIUSRLBL]
;1037     1408  1 !               AND
;1038     1409  1 !               NOT (.HDR1[EO1$L_EO1LID] EQL 'EOF1'
;1039     1410  1 !                   AND
;1040     1411  1 !                   .CURRENT_VCB[VCB$B_TM] EQL 2))
;1041     1412  1 !           THEN
;1042     1413  1 !               ERR_EXIT(SS$_ILLUSRLBLRD);
;1043     1414  1 !
;1044     1415  1 !           ! check if label set terminator has been encountered already
;1045     1416  1 !           !
;1046     1417  1 !           IF (.(.HDR1)<0,16> EQL 'HD'  AND .CURRENT_VCB[VCB$B_TM] EQL 1)
;1047     1418  1 !              OR (.(.HDR1)<0,16> EQL 'EO' AND .CURRENT_VCB[VCB$B_TM] EQL 0)
;1048     1419  1 !           THEN
;1049     1420  1 !               BEGIN
;1050     1421  1 !               ERROR(SS$_ENDOFUSRLBL);     ! no more user labels in the set
;1051     1422  1 !               COUNT = 0;
;1052     1423  1 !               END
;1053     1424  1 !           ELSE
;1054     1425  1 !               BEGIN
;1055     1426  1 !               LOCAL
;1056     1427  1 !                   ID,
;1057     1428  1 !                   SCRATCH : REF BBLOCK;
;1058     1429  1 !               IF .(.HDR1)<0,16> EQL 'HD'
;1059     1430  1 !               THEN
;1060     1431  1 !                   ID = 'UHL'      ! note which set of labels
;1061     1432  1 !               ELSE
;1062     1433  1 !                   ID = 'UTL';     ! are to be read
;1063     1434  1 !
;1064     1435  1 !               ! one label may be in the scratch label area if the was no HDR2
;1065     1436  1 !               !
;1066     1437  1 !               SCRATCH = .HDR1 + 240;      ! address of scratch label
;1067     1438  1 !               IF .(.SCRATCH)<0,24> EQL .ID<0,24>
;1068     1439  1 !               THEN
;1069     1440  1 !                   ! was one read already?
;1070     1441  1 !                   !
;1071     1442  1 !                   BEGIN
;1072     1443  1 !                   CH$MOVE(.SCRATCH,DATA,.COUNT);
;1073     1444  1 !                   .SCRATCH = 0;   ! retreive this one only once
;1074     1445  1 !                   END
;1075     1446  1 !               ELSE
;1076     1447  1 !                   BEGIN           ! no user label previously read or reported
;1077     1448  1 !                   WHILE 1 DO
;1078     1449  1 !                       BEGIN
;1079     1450  1 !                       IF NOT READ_BLOCK(DATA,.COUNT)
;1080     1451  1 !                       THEN
;1081     1452  1 !                           BEGIN
;1082     1453  1 !                           ERROR(SS$_ENDOFUSRLBL);
;1083     1454  1 !                           COUNT = 0;
;1084     1455  1 !                           EXITLOOP;
;1085     1456  1 !                           END;
```

```
;  1086          1457  1 !                                  ! retrieve only valid user labels
;  1087          1458  1 !                                  !
;  1088          1459  1 !                                  IF .DATA<0,24> EQL .ID<0,24>
;  1089          1460  1 !                                  THEN
;  1090          1461  1 !                                      EXITLOOP;
;  1091          1462  1 !                                  END;                ! end of read user label loop
;  1092          1463  1 !                              IF .IO_STATUS<16,16> LSS .COUNT
;  1093          1464  1 !                              THEN
;  1094          1465  1 !                                  COUNT = .IO_STATUS<16,16>;
;  1095          1466  1 !                              USER_STATUS<16,16> = .COUNT;
;  1096          1467  1 !                              END;                ! end of either already read or must read
;  1097          1468  1 !                          END;                    ! end of either end of label set or retrieve label
;  1098          1469  1 !-
;  1099          1470  1
;  1100          1471  1
;  1101          1472  1 END
;  1102          1473  1
;  1103          1474  0 ELUDOM
```

```
;                                  PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $LOCKEDD1$                       4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $DATA$                        1216  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                         894  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        84         0      1000          00:01.9
```

```
;                                     COMMAND QUALIFIERS
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:RDATTR/OBJ=OBJ$:RDATTR MSRC$:RDATTR/UPDATE=(ENH$:RDATTR)

; Size:          882 code + 1232 data bytes
; Run Time:        00:22.5
; Elapsed Time:    00:49.6
; Lines/CPU Min:    3935
; Lexemes/CPU-Min: 17877
```

```
; Memory Used:  180 pages
; Compilation Complete
```

0256 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

WTATTR
LIS

POSFID
LIS

RESACC
LIS

RWVOL
LIS

MTHRTL

STRVIO
LIS

MTHRTL
MAP

POSBE
LIS

RDATTR
LIS

REWSPC
LIS

UUMTHRTL
MAP