

MMM	MMM	TTTTTTTTTTTTTT	AAAAAAAAA	AAAAAAAAA	CCCCCCCCCCCC	PPPPPPPPPPPP	
MMM	MMM	TTTTTTTTTTTTTT	AAAAAAAAA	AAAAAAAAA	CCCCCCCCCCCC	PPPPPPPPPPPP	
MMM	MMM	TTTTTTTTTTTTTT	AAAAAAAAA	AAAAAAAAA	CCCCCCCCCCCC	PPPPPPPPPPPP	
MMMMMM	MMMMMM	TTT	AAA	AAA	CCC	PPP	PPP
MMMMMM	MMMMMM	TTT	AAA	AAA	CCC	PPP	PPP
MMMMMM	MMMMMM	TTT	AAA	AAA	CCC	PPP	PPP
MMM	MMM	TTT	AAA	AAA	CCC	PPP	PPP
MMM	MMM	TTT	AAA	AAA	CCC	PPP	PPP
MMM	MMM	TTT	AAA	AAA	CCC	PPP	PPP
MMM	MMM	TTT	AAA	AAA	CCC	PPP	PPP
MMM	MMM	TTT	AAA	AAA	CCC	PPPPPPPPPPPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPPPPPPPPPPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPPPPPPPPPPP	
MMM	MMM	TTT	AAAAAAAAAAAAAAAA	AAAAAAAAAAAAAAAA	CCC	PPP	
MMM	MMM	TTT	AAAAAAAAAAAAAAAA	AAAAAAAAAAAAAAAA	CCC	PPP	
MMM	MMM	TTT	AAAAAAAAAAAAAAAA	AAAAAAAAAAAAAAAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCCCCCCCCCCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCCCCCCCCCCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCCCCCCCCCCC	PPP	

```
WW      WW      TT/TTTTTTT      AAAAAA      TTTTTTTTTT      TTTTTTTTTT      RRRRRRRR
WW      WW      TTTTTTTTTT      AAAAAA      TTTTTTTTTT      TTTTTTTTTT      RRRRRRRR
WW      WW      TT          AA          AA      TT          TT          RR          RR
WW      WW      TT          AA          AA      TT          TT          RR          RR
WW      WW      TT          AA          AA      TT          TT          RR          RR
WW      WW      TT          AA          AA      TT          TT          RRRRRRRR
WW      WW      TT          AA          AA      TT          TT          RRRRRRRR
WW      WW      TT          AAAAAAAAAA      TT          TT          RR      RR
WW      WW      TT          AAAAAAAAAA      TT          TT          RR      RR
WWWW      WWWW      TT          AA          AA      TT          TT          RR      RR
WWWW      WWWW      TT          AA          AA      TT          TT          RR      RR
WW      WW      TT          AA          AA      TT          TT          RR      RR
WW      WW      TT          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
LLLLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLLLL      IIIIII      SSSSSSSS
```



```
0001 0
0002 0 MODULE WTATTR ( LANGUAGE ( BLISS32 ) ,
0003 0 IDENT = 'V04-000'
0004 0 ) =
0005 1 BEGIN
0006 1
0007 1 *****
0008 1 *
0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0011 1 * ALL RIGHTS RESERVED.
0012 1 *
0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0018 1 * TRANSFERRED.
0019 1 *
0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0022 1 * CORPORATION.
0023 1 *
0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0026 1 *
0027 1 *
0028 1 *****
0029 1
0030 1 ++
0031 1
0032 1 FACILITY: MTAACP
0033 1
0034 1 ABSTRACT:
0035 1 This module handles the writing of attributes on a create
0036 1
0037 1 ENVIRONMENT:
0038 1
0039 1 STARLET operating system, including privileged system services
0040 1 and internal exec routines.
0041 1
0042 1 --
0043 1
0044 1
0045 1 AUTHOR: D. H. GILLESPIE, CREATION DATE: 2-JUN-77
0046 1
0047 1 MODIFIED BY:
0048 1
0049 1 V03-002 LMP0221 L. Mark Pilant, 12-Apr-1984 12:11
0050 1 Change UCBSL_OWNUIC to ORBSL_OWNER and UCBSW_VPROT to
0051 1 ORBSW_PROT.
0052 1
0053 1 V03-002 MMD0278 Meg Dumont, 23-Mar-1984 10:09
0054 1 Change the processing of the accessibility character fields
0055 1 in the HDR1 label to call the installation
0056 1 specific accessibility routine. The return from this
0057 1 routine determines the users access to the file.
```

```
58 0058 1 Fix long file name support such that for ANSI version
59 0059 1 3 volumes it converts the exentsion length to
60 0060 1 ASCII chars before writing it to the label.
61 0061 1
62 0062 1 V03-001 MMD0156 Meg Dumont, 26-Apr-1983 9:16
63 0063 1 Add support fot interchange qualifier. Add support for long
64 0064 1 file name in WANSINAME, by spliting the long name between
65 0065 1 the HDR1 FILE_ID and the HDR4 lables.
66 0066 1
67 0067 1 V02-016 DMW00080 David Michael Walp 2-Mar-1982
68 0068 1 Added check for variable records greater then 9995
69 0069 1
70 0070 1 V02-015 DMW00074 David Michael Walp 5-Feb-1982
71 0071 1 Changed StatBlk to no-op
72 0072 1
73 0073 1 V02-014 DMW00066 David Michael Walp 8-Jan-1982
74 0074 1 Added HDR1 access character support
75 0075 1
76 0076 1 V02-013 DMW00061 David Michael Walp 7-Dec-1981
77 0077 1 Rename TRANSLATION_TABLE to ANSI_A_BAD
78 0078 1
79 0079 1 V02-012 DMW00051 David Michael Walp 10-Nov-1981
80 0080 1 Change SELECTONE to CASE, return length of ASCNAME
81 0081 1 ( not just a flag )
82 0082 1
83 0083 1 V02-011 DMW00046 David Michael Walp 28-Oct-1981
84 0084 1 Added ANSI 'a' character support for file name, reformat
85 0085 1
86 0086 1 V02-010 DMW00042 David Michael Walp 2-Oct-1981
87 0087 1 Added support binary zero input dates
88 0088 1
89 0089 1 V02-009 DMW00029 David Michael Walp 10-Aug-1980
90 0090 1 Added stream file support and removed V2.0 HDR2 temp support
91 0091 1
92 0092 1 V02-008 DMW0004 David Michael Walp 11-Nov-1980
93 0093 1 New BLISS compiler, FUNCTION declaration changed from
94 0094 1 BBLOCK to BLOCK. Old compiler used to give a longword
95 0095 1 with a declartion of 'BBLOCK [1]'.
96 0096 1
97 0097 1 V02-007 REFORMAT Maria del C. Nasr 30-Jun-1980
98 0098 1
99 0099 1 V02-006 MCN0015 Maria del C. Nasr 10-Jun-1980
100 0100 1 Write form control attribute in HDR2 too, and clean
101 0101 1 undefined attributes of HDR3.
102 0102 1
103 0103 1 A0005 MCN0008 Maria del C. Nasr 22-Feb-1980 16:36
104 0104 1 Temporary support of RMS attributes in HDR2
105 0105 1
106 0106 1 A0004 MCN0005 Maria del C. Nasr 05-Nov-1979 11:30
107 0107 1 Eliminate [VCBSV_INTCHG] (interchange) switch
108 0108 1
109 0109 1 A0003 MCN0003 Maria del C. Nasr 16-Oct-1979 14:28
110 0110 1 Add HDR3 processing
111 0111 1
112 0112 1 **
113 0113 1
114 0114 1 LIBRARY 'SYS$LIBRARY:LIB.L32';
```



```
115 0115 1
116 0116 1 REQUIRE 'SRC$:MTADEF.B32';
117 0500 1
118 0501 1 FORWARD ROUTINE
119 0502 1   WANSINAME      : COMMON_CALL NOVALUE,      ! write ANSI 'a' filename
120 0503 1   WBINDATE     : NOVALUE,                  ! write 64_bit date
121 0504 1   WBLOCKSZ     : COMMON_CALL NOVALUE,      ! write block size
122 0505 1   WEXPIREDT    : NOVALUE,                  ! write expiration date
123 0506 1   WRECATTR     : COMMON_CALL NOVALUE,      ! write record attributes
124 0507 1   WRITE_ATTRIBUTE : COMMON_CALL NOVALUE;    ! write attributes, main
125 0508 1
126 0509 1 EXTERNAL
127 0510 1   CURRENT_UCB   : REF BBLOCK,                ! address of UCB
128 0511 1   HDR1          : REF BBLOCK,                ! address of HDR1 label
129 0512 1   HDR2          : REF BBLOCK,                ! address of HDR2 label
130 0513 1   HDR3          : REF BBLOCK,                ! address of HDR3 label
131 0514 1   HDR4          : REF BBLOCK,                ! address of HDR4 label
132 0515 1   NMBLOCK       : BBLOCK [10];              ! name block
133 0516 1
134 0517 1 EXTERNAL ROUTINE
135 0518 1   GET_RECORD,   ! get current record tape is reading
136 0519 1   SYS$FAO      : ADDRESSING_MODE (ABSOLUTE); ! change to hexadecimal
137 0520 1
138 0521 1 OWN
139 0522 1   MRS           : WORD,                        ! maximum record length
140 0523 1   VFCHDR       : WORD,                        ! length of VFC header
141 0524 1   BLOCKSZ;     ! block size
142 0525 1
143 0526 1 BIND
144 0527 1   CVT5 = DESCRIPTOR ('!5ZW'),
145 0528 1   CVT8 = DESCRIPTOR ('!8XL');
146 0529 1
147 0530 1 GLOBAL
148 0531 1   ANSI_NAME_SIZE : SIGNED BYTE;
149 0532 1
```

```
151 0533 1 GLOBAL ROUTINE WRITE_ATTRIBUTE : COMMON_CALL NOVALUE =
152 0534 1
153 0535 1 ++
154 0536 1
155 0537 1 FUNCTIONAL DESCRIPTION:
156 0538 1 This routine writes attributes into HDR1 and HDR2
157 0539 1
158 0540 1 CALLING SEQUENCE:
159 0541 1 WRITE_ATTRIBUTE()
160 0542 1
161 0543 1 INPUT PARAMETERS:
162 0544 1 none
163 0545 1
164 0546 1 IMPLICIT INPUTS:
165 0547 1 IO_PACKET - address of current IO request packet
166 0548 1
167 0549 1 OUTPUT PARAMETERS:
168 0550 1 none
169 0551 1
170 0552 1 IMPLICIT OUTPUTS:
171 0553 1 none
172 0554 1
173 0555 1 ROUTINE VALUE:
174 0556 1 none
175 0557 1
176 0558 1 SIDE EFFECTS:
177 0559 1 none
178 0560 1
179 0561 1 --
180 0562 1
181 0563 2 BEGIN
182 0564 2
183 0565 2 EXTERNAL REGISTER ! here so main expectation
184 0566 2 COMMON_REG; ! handler can touch the VCB
185 0567 2
186 0568 2 EXTERNAL
187 0569 2 ATR_TABLE : BLOCKVECTOR [ATR$C_MAX_CODE, 1], ! attribute table
188 0570 2 IO_PACKET : REF BBLOCK, ! addr of current IRP
189 0571 2 ANSI_A_GOOD : VECTOR [ 256, BYTE ]; ! translation table
190 0572 2
191 0573 2 MACRO EQLLEN = 0, 24, 8, 0%;
192 0574 2 MACRO MAX = 0, 0, 8, 0%;
193 0575 2
194 0576 2 LOCAL
195 0577 2 ACCESS_INPUT, ! set for whether access cahr specified
196 0578 2 CHAR : VECTOR [4,BYTE], ! accessibility char to write to label
197 0579 2 CURRENT_RECORD, ! Record tape drive is reading
198 0580 2 STATUS,
199 0581 2 CHAR_INPUT : BYTE, ! Access char from ATR block
200 0582 2 DESCR : VECTOR [2, LONG], ! descriptor of output for FA0
201 0583 2 MVL : REF BBLOCK, ! MVL of volume
202 0584 2 PACKET : REF BBLOCK, ! IO packet address
203 0585 2 ABD : REF BBLOCKVECTOR [ , ABD$C_LENGTH], ! addr of desc vector
204 0586 2 ORB : REF BBLOCK, ! Object's rights block address
205 0587 2 BITS, ! number of bits set
206 0588 2 P, ! pointer to attribute
207 0589 2 CODE, ! attribute code
```



```

208 0590 2 COUNT; ! number of char in attribute
209 0591
210 0592
211 0593
212 0594
213 0595
214 0596
215 0597
216 0598
217 0599
218 0600
219 0601
220 0602
221 0603
222 0604
223 0605
224 0606
225 0607
226 0608
227 0609
228 0610
229 0611
230 0612
231 0613
232 0614
233 0615
234 0616
235 0617
236 0618
237 0619
238 0620
239 0621
240 0622
241 0623
242 0624
243 0625
244 0626
245 0627
246 0628
247 0629
248 0630
249 0631
250 0632
251 0633
252 0634
253 0635
254 0636
255 0637
256 0638
257 0639
258 0640
259 0641
260 0642
261 0643
262 0644
263 0645
264 0646

PACKET = .IO PACKET;
ABD = .BBLOCK[.PACKET[IRP$S_SVAPTE], AIB$S_DESCRIPTOR];
MRS = 0; ! default record size
BLOCKSZ = 512; ! default record size
VFCHDR = 0; ! initialize VFC header count
ANSI_NAME_SIZE = -1; ! no ANSI file name
ACCESS_INPUT = MTASK_NOCHAR; ! Assume no access char passed

INCRU I FROM ABD$C_ATTRIB TO .PACKET[IRP$W_BCNT] - 1 DO
  BEGIN
    ! set up a pointer to the attribute and set up the size and
    ! find the attribute code

    P = .ABD[I, ABD$W_TEXT] + ABD[I, ABD$W_TEXT];
    COUNT = .ABD[I, ABD$W_COUNT];
    CODE = .(.P) < 0, 8>;
    P = .P + 1;

    ! test for table length
    IF .CODE GTRU ATR$C_MAX_CODE THEN ERR_EXIT(SS$_BADATTRIB);

    ! check attribute length
    IF .ATR_TABLE[(.CODE-1),EQLLEN]
    THEN
      BEGIN
        IF .COUNT NEQU .ATR_TABLE[(.CODE-1),MAX]
        THEN ERR_EXIT(SS$_BADATTRIB);
      END
    ELSE
      BEGIN
        IF .COUNT GTRU .ATR_TABLE[(.CODE-1),MAX]
        THEN ERR_EXIT(SS$_BADATTRIB);
      END;

    ! call the correct procedure
    CASE .CODE FROM 1 TO ATR$C_MAX_CODE OF
      SET
        [ ATR$C_RECATTR ] : WRECATTR ( .P, .COUNT );
        [ ATR$C_FILNAM ] : CH$MOVE ( .COUNT, .P, NMBLOCK );
        [ ATR$C_FILTYP ] : CH$MOVE ( .COUNT, .P, NMBLOCK + 6 );
        [ ATR$C_FILVER ] : CH$MOVE ( .COUNT, .P, NMBLOCK + 8 );
        [ ATR$C_EXPDAT ] : WEXPIREDT ( .P, .COUNT );
        [ ATR$C_BLOCKSIZE ] : WBLOCKSZ ( .P, .COUNT );
        [ ATR$C_ASCNAME ] : WANSINAME ( .P, .COUNT );
        [ ATR$C_CREDATE ] : WBINDATE ( .P, .COUNT,
          BBLOCK[.HDR1,HD1$T_CREATEDT]);
        [ ATR$C_EXPDATE ] : WBINDATE ( .P, .COUNT,
          BBLOCK[.HDR1,HD1$T_EXPIREDT]);
        [ ATR$C_HDR1_ACC ] : BEGIN
```

```

: 265 0647 4
: 266 0648
: 267 0649
: 268 0650
: 269 0651
: 270 0652
: 271 0653
: 272 0654
: 273 0655
: 274 0656
: 275 0657
: 276 0658
: 277 0659
: 278 0660
: 279 0661
: 280 0662
: 281 0663
: 282 0664
: 283 0665
: 284 0666
: 285 0667
: 286 0668
: 287 0669
: 288 0670
: 289 0671
: 290 0672
: 291 0673
: 292 0674
: 293 0675
: 294 0676
: 295 0677
: 296 0678
: 297 0679
: 298 0680
: 299 0681
: 300 0682
: 301 0683
: 302 0684
: 303 0685
: 304 0686
: 305 0687
: 306 0688
: 307 0689
: 308 0690
: 309 0691
: 310 0692
: 311 0693
: 312 0694
: 313 0695
: 314 0696
: 315 0697
: 316 0698
: 317 0699
: 318 0700
: 319 0701
: 320 0702
: 321 0703

CHAR_INPUT = .ANSI_A_GOOD [ (.P)<0,8> ];
ACCESS_INPUT = MTASK_CHARVALID;
END;

! These are either read only attributes or unimplemented ones
[ ATRSC_HEADER,
  ATRSC_USERLABEL,
  ATRSC_ENDLBLAST,
  ATRSC_UIC_RO ] : ERR_EXIT ( SSS_BADATTRIB );

! The following are valid for DISK but not TAPE, so we will drop
! them on the floor ( NO OP ) for device independence sake
! If these values are asked for during read attributes we will
! dummy them up
! INRANGE statement is used for speed and code size sake
[ 1,
  2,
  ATRSC_UCHAR,
  ATRSC_STATBLK,
  ATRSC_ASCDATES,
  ATRSC_ALCONTROL,
  ATRSC_REVDATE,
  ATRSC_BAKDATE,
  ATRSC_UIC,
  ATRSC_FPRO,
  ATRSC_RPRO,
  ATRSC_ACLEVEL,
  ATRSC_SEMASK,
  ATRSC_DIRSEQ,
  ATRSC_BACKLINK,
  ATRSC_JOURNAL ] : ;
[ INRANGE ] : ;
TES;
END;

! Call the accessibility system service to determine the accessibility char
! to write on the label.
! First keep the record that the UCB is reading. The accessibility
! routine can not move the tape from under us! Thus we will compare
! this to the field after the call and if the tape was moved we punt
! the operation.

MVL = .CURRENT_VCB[VCBSL_MVL];
ORB = .CURRENT_UCB[UCBSL_ORB];
CURRENT_RECORD = KERNEL_CALL(GET_RECORD, .CURRENT_UCB);
CHAR = $MTACCESS(LBLNAM = .HDR1,
                  UIC = .ORB[ORB$OWNER],
                  STD_VERSION = .MVL[MVL$STDVER],
                  ACCESS_CHAR = .CHAR_INPUT,
                  ACCESS_SPEC = .ACCESS_INPUT,
                  TYPE = .MTASK_OUTHDR1);

STATUS = KERNEL_CALL(GET_RECORD, .CURRENT_UCB);
IF .CURRENT_RECORD NEQ .STATUS
  THEN ERR_EXIT(SSS_TAPEPOSLOST);
```



```
! Write the char to the label.
HDR1[HD1$B_FILACCESS] = .CHAR[0];
! after both blocksize and maximum record size attributes have been
! processed then determine the value of reclen in HDR2
IF .MRS EQLU 0
THEN
  (CH$MOVE(HD2$S_BLOCKLEN, HDR2[HD2$T_BLOCKLEN], HDR2[HD2$T_RECLEN]))
ELSE
  BEGIN
    ! count overhead for ANSI 'D' record
    IF .HDR2[HD2$B_RECFORMAT] EQL 'D' THEN MRS = .MRS + 4 + .VFCHDR;
    ! FAO output descriptor
    DESCR[0] = HD2$S_RECLEN;
    DESCR[1] = HDR2[HD2$T_RECLEN];
    SYS$FAO(CVT5, 0, DESCR, .MRS);
  END;
! now check
! that the block size is valid
! and that the record size is not bigger then the block size
! and that variable records are not larger then can be supported ( RCW )
IF ( .BLOCKSZ LSSU 18 )
OR ( .MRS GTRU .BLOCKSZ )
OR ( ( .MRS GTRU (9999 - 4) ) AND ( .HDR2 [ HD2$B_RECFORMAT ] EQL 'D' ) )
THEN ERR_EXIT(SS$_BADATTRIB);
END;
```

```
.TITLE WTATTR
.IDENT \V04-000\
```

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

```
57 5A 35 21 00000 P.AAB: .ASCII \!5ZW\
      00000004 00004 P.AAA: .LONG 4
      00000000' 00008 .ADDRESS P.AAB
4C 58 38 21 0000C P.AAD: .ASCII \!8XL\
      00000004 00010 P.AAC: .LONG 4
      00000000' 00014 .ADDRESS P.AAD
```

```
.PSECT $LOCKEDD1$,NOEXE,2
```

```
00000 MRS: .BLKB 2
00002 VFCHDR: .BLKB 2
00004 BLOCKSZ: .BLKB 4
00008 ANSI_NAME_SIZE::
      .BLKB 1
```

CVT5=
CVT8=P.AAA
P.AAC

.EXTRN CURRENT_UCB, HDR1
.EXTRN HDR2, HDR3, HDR4
.EXTRN NMBLOCK, GET_RECORD
.EXTRN SYSSFAO, ATR_TABLE
.EXTRN IO_PACKET, ANSI_A_GOOD
.EXTRN SYSSCMKRNL, SYSSMTACCESS

.PSECT \$CODE\$,NOWRT,2

.ENTRY WRITE_ATTRIBUTE, Save R2,R3,R4,R5,R6,R7,R8,-; 0533

SUBL2 #8, SP
MOVL IO_PACKET, PACKET 0593
MOVL @4(PACKET), ABD 0594
MOVZWL #512, BLOCKSZ 0596
CLRL MRS 0595
MNEGB #1, ANSI_NAME_SIZE 0598
CLRL ACCESS_INPUT 0599
MOVZWL 50(PACKET), -(SP) 0601
DECL (SP)

MOVL #5, I
BSBW 20\$
MOVAQ (ABD)[I], R0 0607
MOVZWL (R0), P
ADDL2 R0, P

PUSHAQ 2(ABD)[I] 0608
MOVZWL @ (SP)+, COUNT
MOVZBL (P)+, CODE 0609
CMLP CODE, #48 0614
BLEQU 2\$

CHMU #52
BLBC ATR_TABLE-1[CODE], 3\$ 0618
PUSHAL ATR_TABLE-4[CODE] 0621
CMPZV #0, #8, @ (SP)+, COUNT

BEQL 5\$
BRB 4\$ 0622
PUSHAL ATR_TABLE-4[CODE] 0626
CMPZV #0, #8, @ (SP)+, COUNT

BGEQU 5\$
CHMU #52 0627
CASEL CODE, #1, #47 0632
.WORD 19\$-6\$,-

19\$-6\$,-
19\$-6\$,-
19\$-6\$,-
7\$-6\$,-
8\$-6\$,-
9\$-6\$,-
10\$-6\$,-
11\$-6\$,-
19\$-6\$,-
18\$-6\$,-
12\$-6\$,-
18\$-6\$,-
19\$-6\$,-

07FC 00000

5E 08 C2 00002
50 0000G CF D0 00005
5A 2C B0 D0 0000A
0000' CF 0200 8F 3C 0000E
0000' CF 0000' CF D4 00015
CF 01 8E 00019
7E 7E D4 0001E
32 A0 3C 00020
57 6E D7 00024
05 D0 00026
010C 30 00029
50 6A47 7E 0002C 1\$:
58 60 3C 00030
58 50 C0 00033
02 AA47 7F 00036
59 9E 3C 0003A
56 88 9A 0003D
30 56 D1 00040
02 1B 00043
34 BF 00045
0E 0000GCF46 E9 00047 2\$:
08 0000GCF46 DF 0004D
00 ED 00052
10 13 00057
0C 11 00059
0000GCF46 DF 0005B 3\$:
08 00 ED 00060
02 1E 00065
34 BF 00067 4\$:
56 CF 00069 5\$:
0006D 6\$:

0060 00C9 00C9 00C9 0006D
0082 007A 0072 006A 00075
00C7 008C 00C7 00C9 0007D
0096 00C7 00C9 00C9 00085
00C9 00A8 00C9 00A0 0008D
00C9 00C9 00C9 00C9 00095
00C9 00C9 00C7 00C9 0009D
00C9 00C9 00B8 00C9 000A5
00C9 00C9 00C9 00C9 000AD
00C9 00C9 00C9 00C9 000B5
00C9 00C9 00C9 00C9 000BD
00C9 00C9 00C9 00C9 000C5

Address	Op Code	Op Name	Comment	Hex
0000G	CF	68	59 28 000D7 8\$:	MOVQ P, -(SP)
0000G	CF	68	57 11 000DD	CALLS #2, WRECATTR
0000G	CF	68	59 28 000DF 9\$:	BRB 19\$
0000G	CF	68	4F 11 000E5	MOVQ P, -(SP)
			59 28 000E7 10\$:	CALLS #2, WEXPIREDT
			47 11 000ED	BRB 19\$
			58 7D 000EF 11\$:	MOVQ P, -(SP)
			02 FB 000F2	CALLS #2, WBLOCKSZ
			3D 11 000F7	BRB 19\$
			58 7D 000F9 12\$:	MOVQ P, -(SP)
			02 FB 000FC	CALLS #2, WANSINAME
			33 11 00101	BRB 19\$
			58 7D 00103 13\$:	MOVQ P, -(SP)
			02 FB 00106	CALLS #41, HDR1, -(SP)
			29 11 0010B	BRB 16\$
7E	0000G	CF	29 C1 0010D 14\$:	ADDL3 #47, HDR1, -(SP)
7E	0000G	CF	06 11 00113	BRB
			2F C1 00115 15\$:	ADDL3
			58 7D 0011B 16\$:	MOVQ P, -(SP)

0000V	CF	03	FB	0011E	CALLS	#3, WBINDATE	:	
		11	11	00123	BRB	19\$	:	
	50	68	9A	00125	MOVZBL	(P), R0	:	0647
	6E	0000GCF	40	90	00128	MOVB	ANSI_A_GOOD[R0],	
08	AE		01	D0	0012E	#1, ACCESS_INPUT	:	0648
			02	11	00132	BRB	:	0632
			34	BF	00134	CHMU	:	0656
			57	D6	00136	I	:	0601
04	AE		57	D1	00138	CMPL	:	
			03	1A	0013C	I, 4(SP)	:	
			03	1A	0013C	BGTRU	:	
			31	0013E	BRW	1\$	:	
	52	34	AB	D0	00141	21\$:	:	0691
	50	0000G	CF	D0	00145	MOVL	:	0692
	53	1C	A0	D0	0014A	MOVL	:	
			50	DD	0014E	MOVL	:	0693
			01	DD	00150	PUSHL	:	
			5E	DD	00152	PUSHL	:	
			CF	9F	00154	PUSHL	:	
00000000G	9F	0000G	04	FB	00158	PUSHAB	:	
	54		50	D0	0015F	GET_RECORD	:	
			03	DD	00162	CALLS	:	
			0C	AE	00164	#4, @#SYSS\$CMKRNL	:	0699
	7E		08	AE	00167	R0, CURRENT_RECORD	:	
	7E		22	A2	0016B	PUSHL	:	
			63	DD	0016F	#3	:	
			CF	DD	00171	ACCESS INPUT	:	
00000000G	00	0000G	06	FB	00175	MOVZBL	:	
	52		50	D0	0017C	CHAR INPUT, -(SP)	:	
			CF	DD	0017F	MOVZBL	:	
			01	DD	00183	34(MVL), -(SP)	:	
			5E	DD	00185	PUSHL	:	
			CF	9F	00187	PUSHL	:	
00000000G	9F	0000G	04	FB	0018B	HDR1	:	
	50		54	D1	00192	CALLS	:	
			04	13	00195	#6, SYSS\$MTACCESS	:	
			8F	BF	00197	R0, CHAR	:	
	50	0000G	CF	D0	0019B	MOVZBL	:	
35	A0		52	90	001A0	22\$:	:	0701
	56	0000G	CF	D0	001A4	MOVL	:	
	57	0000'	CF	3C	001A9	MOVB	:	
			08	12	001AE	MOVL	:	
0A	A6		05	28	001B0	MOVZWL	:	
			34	11	001B6	MRS, R7	:	
			0F	12	001BD	BNEQ	:	
	44		A6	91	001B8	23\$:	:	0714
			0F	12	001BD	MOVZWL	:	0712
	50	0000'	CF	3C	001BF	MOVZWL	:	
	51	04	A740	9E	001C4	VFCHDR, R0	:	
	OC		51	B0	001C9	4(R7)[R0], R1	:	
	10		05	D0	001CE	MOVW	:	
			0A	A6	001D2	R1, MRS	:	
			CF	3C	001D7	24\$:	:	0724
			7E	AE	001DC	MOVL	:	0725
			7E	D4	001DF	MOVZWL	:	0726
			CF	9F	001E1	MRS, -(SP)	:	
			04	FB	001E5	PUSHAB	:	
00000000G	9F	FE07	CF	9F	001E1	DESCR	:	
	12	0000'	CF	D1	001EC	CLRL	:	
			20	1F	001F1	-(SP)	:	
						CVT5	:	
						CALLS	:	
						#4, @#SYSS\$FAO	:	
						CMPL	:	
						BLSSU	:	
						26\$	:	0734

WTATTR
V04-000

M 11
16-Sep-1984 02:33:32
14-Sep-1984 12:46:51

VAX-11 Bliss-32 V4.0-742
[MTAACP.SRC]WTATTR.B32;1

Page 11
(2)

0000' CF 0000' CF

10
270B 8F

0000'

50
44 8F

0000G
04

00 ED 001F3
15 1A 001FC
CF B1 001FE
OE 1B 00205
CF D0 00207
A0 91 0020C
02 12 00211
34 BF 00213 26\$:
04 00215 27\$:

CMPZV #0, #16, MRJ, BLOCKSZ
BGTRU 26\$
CMPW MRS, #9995
BLEQU 27\$
MOVL HDR2, R0
CMPB 4(R0), #68
BNEQ 27\$
CHMU #52
RET

: 0735
: 0736
: 0737
: 0739

; Routine Size: 534 bytes, Routine Base: \$CODE\$ + 0018

```

359 0740 1 ROUTINE WEXPIREDT (PTER, COUNT) : NOVALUE =
360 0741 1
361 0742 1 ++
362 0743 1
363 0744 1 FUNCTIONAL DESCRIPTION:
364 0745 1 This routine puts the expiration attribute in HDR1
365 0746 1
366 0747 1 CALLING SEQUENCE:
367 0748 1 WEXPIREDT(ARG1,ARG2)
368 0749 1
369 0750 1 INPUT PARAMETERS:
370 0751 1 ARG1 - address of attribute(ddmmmyy)
371 0752 1 ARG2 - number of characters in attribute
372 0753 1
373 0754 1 IMPLICIT INPUTS:
374 0755 1 none
375 0756 1
376 0757 1 OUTPUT PARAMETERS:
377 0758 1 none
378 0759 1
379 0760 1 IMPLICIT OUTPUTS:
380 0761 1 expiration date in HDR1 updated
381 0762 1
382 0763 1 ROUTINE VALUE:
383 0764 1 none
384 0765 1
385 0766 1 SIDE EFFECTS:
386 0767 1 none
387 0768 1
388 0769 1 USER ERRORS:
389 0770 1 $$$_BADATTRIB - attribute length must equal 7 or illegal date
390 0771 1
391 0772 1 --
392 0773 1
393 0774 2 BEGIN
394 0775 2
395 0776 2 EXTERNAL
396 0777 2 ZERO_JDATE : VECTOR [ 6, BYTE ];! zero Julian Date of " 00000"
397 0778 2
398 0779 2 EXTERNAL ROUTINE
399 0780 2 CONVDATE_R2J; ! convert date from reg to JULIAN(ANSI)
400 0781 2
401 0782 2 LOCAL
402 0783 2 DATE : VECTOR [12, BYTE]; ! work area for date with dashes
403 0784 2
404 0785 3 IF (.(.PTER) EQL 0) AND (.(.PTER+4) EQL 0)
405 0786 2 THEN
406 0787 2
407 0788 2 ! if expiration is binary zero, then no date thus stuff zero date
408 0789 2
409 0790 2 (CH$MOVE ( 6, ZERO_JDATE, BBLOCK[.HDR1, HD1$T_EXPIREDT] ))
410 0791 2
411 0792 2 ELSE
412 0793 2 BEGIN
413 0794 2
414 0795 2 ! now format DD-MMM-YYYY from DDMMYY
415 0796 2
```



```
: 416      0797      3      (DATE)<0, 16> = .(.PTER);
: 417      0798      3      (DATE + 2)<0, 8> = '-';
: 418      0799      3      (DATE + 3)<0, 24> = .(.PTER + 2);
: 419      0800      3      (DATE + 6)<0, 24> = '-19';
: 420      0801      3      (DATE + 9)<0, 16> = .(.PTER + 5);
: 421      0802      3
: 422      0803      3
: 423      0804      3
: 424      0805      3
: 425      0806      3
: 426      0807      3
: 427      0808      1

      IF NOT CONVDATE_R2J(DATE, BBLOCK[.HDR1, HD1$T_EXPIREDT])
      THEN
        ERR_EXIT(SS$ _BADATTRIB);
      END;
END;
```

.EXTRN ZERO_JDATE, CONVDATE_R2J

007C 00000 WEXPIREDT:

			5E		0C	C2	00002		.WORD	Save R2,R3,R4,R5,R6	: 0740
			56	04	AC	D0	00005		SUBL2	#12, SP	: 0785
					66	D5	00009		MOVL	PTER, R6	
					12	12	0000B		TSTL	(R6)	
				04	A6	D5	0000D		BNEQ	1\$	
					0D	12	00010		TSTL	4(R6)	
			50	0000G	CF	D0	00012		BNEQ	1\$	: 0790
	2F	A0			06	28	00017		MOVL	HDR1, R0	
						04	0001E		MOVC3	#6, ZERO_JDATE, 47(R0)	
			6E		66	B0	0001F	1\$:	RET		
03	AE		02	AE	2D	90	00022		MOVW	(R6), DATE	: 0797
				00	A6	F0	00026		MOVB	#45, DATE+2	: 0798
					09	AE	9F	0002D	INSV	2(R6), #0, #24, DATE+3	: 0799
		0039312D	9F		01	FB	00030		PUSHAB	DATE+9	: 0801
			51		05	A6	D0	00037	CALLS	#1, @#^X0039312D	
			60			51	B0	0003B	MOVL	5(R6), R1	
06	AE		00		51	F0	0003E		MOVW	R1, (R0)	
		7E			2F	C1	00044		INSV	R1, #0, #24, DATE+6	
		0000G	CF		04	AE	9F	0004A	ADDL3	#47, HDR1, -(SP)	: 0803
						02	FB	0004D	PUSHAB	DATE	
		0000G	CF			50	E8	00052	CALLS	#2, CONVDATE_R2J	
			02			34	BF	00055	BLBS	R0, 2\$	: 0805
						04	00057	2\$:	CHMU	#52	: 0808
									RET		

; Routine Size: 88 bytes, Routine Base: \$CODE\$ + 022E

```

: 429 0809 1 ROUTINE WRECATTR (PTER, COUNT) : COMMON_CALL NOVALUE =
: 430 0810 1
: 431 0811 1 ++
: 432 0812 1
: 433 0813 1 FUNCTIONAL DESCRIPTION:
: 434 0814 1 This routine records user attributes in HDR2 and HDR3
: 435 0815 1
: 436 0816 1 CALLING SEQUENCE:
: 437 0817 1 WRECATTR(ARG1,ARG2)
: 438 0818 1
: 439 0819 1 INPUT PARAMETERS:
: 440 0820 1 ARG1 - address of attribute
: 441 0821 1 ARG2 - size of attribute
: 442 0822 1
: 443 0823 1 IMPLICIT INPUTS:
: 444 0824 1 none
: 445 0825 1
: 446 0826 1 OUTPUT PARAMETERS:
: 447 0827 1 none
: 448 0828 1
: 449 0829 1 IMPLICIT OUTPUTS:
: 450 0830 1 HDR2 and HDR3 updated to reflect user attributes
: 451 0831 1
: 452 0832 1 ROUTINE VALUE:
: 453 0833 1 none
: 454 0834 1
: 455 0835 1 SIDE EFFECTS:
: 456 0836 1 none
: 457 0837 1
: 458 0838 1 --
: 459 0839 1
: 460 0840 2 BEGIN
: 461 0841 2
: 462 0842 2 EXTERNAL REGISTER
: 463 0843 2 COMMON_REG; ! Set up reference to the VCB
: 464 0844 2
: 465 0845 2 LITERAL
: 466 0846 2 RECTYPE_LEN = 8,
: 467 0847 2 RECATTR_LEN = 6;
: 468 0848 2
: 469 0849 2 BIND
: 470 0850 2 RECTYPE_ANSI = UPLIT BYTE('UFDVDDDS') : VECTOR [, BYTE],
: 471 0851 2 RECTYPE_FILE11 = UPLIT BYTE(0,1,2,3,4,5,6,0) : VECTOR [, BYTE],
: 472 0852 2 RECATTR_ASCII = UPLIT BYTE('MA MA ') : VECTOR [, BYTE],
: 473 0853 2 RECATTR_FILE11 = UPLIT BYTE(0, 1, 2, 8, 9, 10) : VECTOR [, BYTE];
: 474 0854 2
: 475 0855 2 LOCAL
: 476 0856 2 ATTRIBUTES : VECTOR [CH$ALLOCATION(32)], ! store good attributes
: 477 0857 2 X, ! for support of HDR2 attributes
: 478 0858 2 DESCR : VECTOR [2], ! general descriptor
: 479 0859 2 PARTIAL_CNT; ! count of attributes to transfer
: 480 0860 2
: 481 0861 2 MAP
: 482 0862 2 PTER : REF BBLOCK; ! address of attribute
: 483 0863 2
: 484 0864 2 ! capture valid attributes only
: 485 0865 2
```



```

: 486      0866 2  CH$COPY(.COUNT, .PTER, 0, 32, ATTRIBUTES);
: 487      0867 2
: 488      0868 2  ! is record type present
: 489      0869 2
: 490      0870 3  IF .COUNT GEQU ($BYTEOFFSET(FAT$B_RTYPE) + 1)
: 491      0871 3  THEN
: 492      0872 3      BEGIN
: 493      0873 3      HDR2[HD2$B_RECFORMAT] = %C'U';
: 494      0874 3
: 495      0875 3      DECR I FROM RECTYPE_LEN - 1 TO 0 DO
: 496      0876 4      BEGIN
: 497      0877 4          IF .RECTYPE_FILE11[I] EQL .PTER[FAT$B_RTYPE]
: 498      0878 4          THEN
: 499      0879 4              HDR2[HD2$B_RECFORMAT] = .RECTYPE_ANSI[I];
: 500      0880 4
: 501      0881 4      END;
: 502      0882 3
: 503      0883 3  IF .HDR2[HD2$B_RECFORMAT] EQLU 'V'
: 504      0884 3  THEN
: 505      0885 3      BEGIN
: 506      0886 4      HDR2[HD2$B_RECFORMAT] = 'D';
: 507      0887 4
: 508      0888 4      IF .COUNT GEQU ($BYTEOFFSET(FAT$B_VFCSIZE) + 1)
: 509      0889 5      THEN
: 510      0890 4          VFCHDR = .PTER[FAT$B_VFCSIZE]      ! get VFC header size
: 511      0891 4      ELSE
: 512      0892 4          VFCHDR = 2;                        ! if defined,
: 513      0893 4          ! otherwise, set default value
: 514      0894 4
: 515      0895 3      END;
: 516      0896 3
: 517      0897 3  ! If this tape is being created for interchange do not allow VMS
: 518      0898 3  ! specific values in this field
: 519      0899 3
: 520      0900 3  IF .CURRENT_VCB[VCB$V_INTCHG] AND .HDR2[HD2$B_RECFORMAT] EQLU 'U'
: 521      0901 3  THEN
: 522      0902 3      ERR_EXIT(SS$_BADATTRIB);
: 523      0903 3
: 524      0904 3  END;
: 525      0905 3
: 526      0906 3  ! if form control character is present
: 527      0907 3
: 528      0908 3  IF .COUNT GEQU ($BYTEOFFSET(FAT$B_RATTRIB) + 1)
: 529      0909 3  THEN
: 530      0910 3      BEGIN
: 531      0911 3
: 532      0912 3      INCR I FROM 0 TO RECATTR_LEN - 1 DO
: 533      0913 3
: 534      0914 3          IF .RECATTR_FILE11[I] EQLU .PTER[FAT$B_RATTRIB]
: 535      0915 3          THEN
: 536      0916 3              HDR2[HD2$B_FORMCNTRL] = .RECATTR_ASCII[I];
: 537      0917 3
: 538      0918 3      END;
: 539      0919 3
: 540      0920 3  ! copy record attributes into HDR3 in hexadecimal format...
: 541      0921 3
: 542      0922 2  PARTIAL_CNT = .COUNT - 1;
```

```

543 0923 2
544 0924 2
545 0925 2
546 0926 2
547 0927 2
548 0928 2
549 0929 2
550 0930 2
551 0931 2
552 0932 2
553 0933 2
554 0934 2
555 0935 2
556 0936 2
557 0937 2
558 0938 2
559 0939 2
560 0940 2
561 0941 2
562 0942 2
563 0943 2
564 0944 2
565 0945 2
566 0946 2
567 0947 2
568 0948 2
569 0949 2
570 0950 2
571 0951 2
572 0952 2
573 0953 2
574 0954 2
575 0955 2
576 0956 2
577 0957 2
578 0958 2
579 0959 2
580 0960 2
581 0961 2
582 0962 2
583 0963 1

IF .PARTIAL_CNT GTR 28
THEN
    PARTIAL_CNT = 28;

DESCR[0] = 8;

INCR I FROM 0 TO .PARTIAL_CNT BY 4 DO
    BEGIN
        DESCR[1] = HDR3[HD3$T_RECATR] + (.I*2);
        SY$FAO(CVT8, 0, DESCR, .(ATTRIBUTES + .I));
    END;

! calculate maximum record size
MRS = .(ATTRIBUTES + $BYTEOFFSET(FAT$W_RSIZE))<0, 16>;

IF .MRS EQL 0
THEN
    MRS = .(ATTRIBUTES + $BYTEOFFSET(FAT$W_MAXREC))<0, 16>;

! Copy attributes to HDR2 too, for temporary support
X = 1;

IF NOT .X
THEN
    BEGIN
        PARTIAL_CNT = .COUNT;

        IF .PARTIAL_CNT GTRU 20
        THEN
            PARTIAL_CNT = 20;

        CH$COPY(.PARTIAL_CNT, .PTER, 0, HD2$$_RECATR1, HDR2[HD2$T_RECATR1]);
        PARTIAL_CNT = .COUNT - .PARTIAL_CNT;
        CH$COPY(.PARTIAL_CNT, .PTER + HD2$$_RECATR1, 0, HD2$$_RECATR2,
            HDR2[HD2$T_RECATR2]);
    END;

END;
```

```

53 44 44 44 56 44 46 55 00286 P.AAE: .ASCII \UFDVDDDS\
00 06 05 04 03 02 01 00 0028E P.AAF: .BYTE 0, 1, 2, 3, 4, 5, 6, 0
      20 41 4D 20 41 4D 00296 P.AAG: .ASCII \MA MA \
      0A 09 08 02 01 00 0029C P.AAH: .BYTE 0, 1, 2, 8, 9, 10
```

```

RECTYPE_ANSI= P.AAE
RECTYPE_FILE11= P.AAF
RECATTR_ASCII= P.AAG
RECATTR_FILE11= P.AAH
```

```

00FC 00000 WRECATTR:
               .WORD Save R2,R3,R4,R5,R6,R7
```

; 0809

20	00	57	0000'	CF	9E	00002	MOVAB	VFCHDR, R7	
		5E		28	C2	00007	SUBL2	#40, SP	
		56	04	AC	D0	0000A	MOVL	PTER, R6	0866
		66	08	AC	2C	0000E	MOVCS	COUNT, (R6), #0, #32, ATTRIBUTES	
			08	AE		00014			
			08	AC	D5	00016	TSTL	COUNT	0870
				45	13	00019	BEQL	5\$	
		50	0000G	CF	D0	0001B	MOVL	HDR2, R0	0873
		51	04	A0	9E	00020	MOVAB	4(R0), R1	
		61	55	8F	90	00024	MOVB	#85, (R1)	
		50		07	D0	00028	MOVL	#7, I	0875
		66	BD	AF40	91	0002B	CMPB	RECTYPE_FILE11[I], (R6)	0878
				05	12	00030	BNEQ	2\$	
		61	AE	AF40	90	00032	MOVB	RECTYPE_ANSI[I], (R1)	0880
		F1		50	F4	00037	SOBGEQ	I, 1\$	0875
56		8F		61	91	0003A	CMPB	(R1), #86	0884
				13	12	0003E	BNEQ	4\$	
		61	44	8F	90	00040	MOVB	#68, (R1)	0887
		10	08	AC	D1	00044	CMPL	COUNT, #16	0889
				06	1F	00048	BLSSU	3\$	
		67	0F	A6	9B	0004A	MOVZBW	15(R6), VFCHDR	0891
				03	11	0004E	BRB	4\$	
		67		02	B0	00050	MOVW	#2, VFCHDR	0893
08	2C	AB		04	E1	00053	BBC	#4, 44(CURRENT_VCB), 5\$	0900
	55	8F		61	91	00058	CMPB	(R1), #85	
				02	12	0005C	BNEQ	5\$	
				34	BF	0005E	CHMU	#52	0902
		02	08	AC	D1	00060	CMPL	COUNT, #2	0908
				1E	1F	00064	BLSSU	8\$	
		52	04	AC	D0	00066	MOVL	PTER, R2	0914
				50	D4	0006A	CLRL	I	
		01	A2	8A	AF40	91	CMPB	RECATR_FILE11[I], 1(R2)	
				0C	12	00072	BNEQ	7\$	
		51	0000G	CF	D0	00074	MOVL	HDR2, R1	0916
	24	A1	FF76	CF40	90	00079	MOVB	RECATR_ASCII[I], 36(R1)	
		50		05	F3	00080	AOBLEQ	#5, I, 8\$	0914
E8		08		01	C3	00084	SUBL3	#1, COUNT, PARTIAL_CNT	0922
53		1C		53	D1	00089	CMPL	PARTIAL_CNT, #28	0924
				03	15	0008C	BLEQ	9\$	
		53		1C	D0	0008E	MOVL	#28, PARTIAL_CNT	0926
		6E		08	D0	00091	MOVL	#8, DESCR	0928
		52		04	CE	00094	MNEGL	#4, I	0930
				21	11	00097	BRB	11\$	
	04	AE	0000G	D42	3E	00099	MOVW	@HDR3[I], DESCR+4	0932
	04	AE		04	C0	000A0	ADDL2	#4, DESCR+4	
			08	AE42	9F	000A4	PUSHAB	ATTRIBUTES[I]	0933
				9E	DD	000A8	PUSHL	@(SP)+	
			04	AE	9F	000AA	PUSHAB	DESCR	
				7E	D4	000AD	CLRL	-(SP)	
			FCBB	CF	9F	000AF	PUSHAB	CVT8	
		9F		04	FB	000B3	CALLS	#4, @#SYSSFA0	
FFD9	52	04	00000000G	53	F1	000BA	ACBL	PARTIAL_CNT, #4, I, 10\$	0930
		FE		0A	AE	000C0	MOVW	ATTRIBUTES+2, MRS	0938
				05	12	000C5	BNEQ	12\$	0940
		FE		18	AE	000C7	MOVW	ATTRIBUTES+16, MRS	0942
				04	000CC	12\$:	RET		0963

WTATTR
V04-000

6 12
16-Sep-1984 02:33:32
14-Sep-1984 12:46:51

VAX-11 Bliss-32 V4.0-742
[MTAACP.SRC]WTATTR.B32;1

Page 18
(4)

; Routine Size: 205 bytes, Routine Base: \$CODE\$ + 02A2


```

585 0964 1 ROUTINE WBLOCKSZ (ADDR, LENGTH) : COMMON_CALL NOVALUE =
586 0965 1
587 0966 1 ++
588 0967 1
589 0968 1 FUNCTIONAL DESCRIPTION:
590 0969 1 This routine writes the block size
591 0970 1
592 0971 1 CALLING SEQUENCE:
593 0972 1 WBLOCKSZ(ARG1,ARG2)
594 0973 1
595 0974 1 INPUT PARAMETERS:
596 0975 1 ARG1 - address of attribute
597 0976 1 ARG2 - length of attribute
598 0977 1
599 0978 1 IMPLICIT INPUTS:
600 0979 1 none
601 0980 1
602 0981 1 OUTPUT PARAMETERS:
603 0982 1 none
604 0983 1
605 0984 1 IMPLICIT OUTPUTS:
606 0985 1 block size written to HDR2
607 0986 1
608 0987 1 ROUTINE VALUE:
609 0988 1 none
610 0989 1
611 0990 1 SIDE EFFECTS:
612 0991 1 none
613 0992 1
614 0993 1 USER ERRORS:
615 0994 1 SSS_BADATTRIB - block size attribute neq to 2
616 0995 1 SSS_NOTINTBLSZ - block size on interchange tape not <=2048
617 0996 1 --
618 0997 1
619 0998 2 BEGIN
620 0999 2
621 1000 2 EXTERNAL REGISTER
622 1001 2 COMMON_REG; ! Set up reference to the VCB
623 1002 2
624 1003 2 MAP
625 1004 2 ADDR : REF VECTOR [, BYTE]; ! address of attribute
626 1005 2
627 1006 2 LOCAL
628 1007 2 DESCR : VECTOR [2];
629 1008 2
630 1009 2 BLOCKSZ = .(.ADDR)<0, 16>;
631 1010 2
632 1011 2 ! If this tape is being created for interchange do not allow VMS
633 1012 2 ! specific values in this field
634 1013 2
635 1014 2 IF .CURRENT_VCB[VCB$V_INTCHG]
636 1015 2 THEN
637 1016 2 IF .BLOCKSZ GTRU 2048
638 1017 2 THEN
639 1018 2 ERR_EXIT (SSS_NOTINTBLSZ);
640 1019 2
641 1020 2 DESCR[0] = HD2$$_BLOCKLEN;
```

WTATTR
V04-000

I 12
16-Sep-1984 02:33:32
14-Sep-1984 12:46:51

VAX-11 Bliss-32 V4.0-742
[MTAACP.SRC]WTATTR.B32;1

Page 20
(5)

```
: 642      1021  2  DESCR[1] = HDR2[HD2$T,BLOCKLEN];
: 643      1022  2  SYSS$FAO(CVT5, 0, DESCR, .BLOCKSZ);
: 644      1023  1  END;
```

```
                                0004 00000 WBLOCKSZ:
                                52      0000' CF 9E 00002 .WORD      Save R2
                                5E      08 C2 00007 MOVAB      BLOCKSZ, R2
                                62      04 BC 3C 0000A SUBL2     #8, SP
                                OD      04 04 E1 0000E MOVZWL    @ADDR, BLOCKSZ
                                2C      62 D1 00013 BBC      #4, 44(CURRENT_VCB), 1$
                                00000800 8F 04 1B 0001A CMPL     BLOCKSZ, #2048
                                01D4    8F BF 0001C BLEQU     1$
                                04 AE 0000G 6E 05 D0 00020 1$: CHMU     #468
                                CF      05 C1 00023 MOVL     #5, DESCR
                                04      62 DD 0002A ADDL3    #5, HDR2, DESCR+4
                                04      7E D4 0002F PUSHL    BLOCKSZ
                                FC60    CF 9F 00031 PUSHAB   DESCR
                                04      04 FB 00035 CLRL     -(SP)
                                9F      04 04 0003C CALLS    CVT5
                                00000000G 9F 04 FB 00035 CALLS    #4, @SYSS$FAO
                                RET
                                : 0964
                                : 1009
                                : 1014
                                : 1016
                                : 1018
                                : 1020
                                : 1021
                                : 1022
                                :
                                : 1023
```

; Routine Size: 61 bytes, Routine Base: \$CODE\$ + 036F

```
: 645      1024  1
```



```

: 647 1025 1 ROUTINE WBINDATE (INPUT, INPLEN, OUTPUT) : NOVALUE =
: 648 1026 1
: 649 1027 1 ++
: 650 1028 1
: 651 1029 1 FUNCTIONAL DESCRIPTION:
: 652 1030 1 This routine converts a 64 bit input date to an ANSI Julian date
: 653 1031 1
: 654 1032 1 CALLING SEQUENCE:
: 655 1033 1 WBINDATE(ARG1,ARG2,ARG3)
: 656 1034 1
: 657 1035 1 INPUT PARAMETERS:
: 658 1036 1 ARG1 - address of 64 bit date
: 659 1037 1 ARG2 - length of date
: 660 1038 1 ARG3 - address for Julian date to placed
: 661 1039 1
: 662 1040 1 IMPLICIT INPUTS:
: 663 1041 1 none
: 664 1042 1
: 665 1043 1 OUTPUT PARAMETERS:
: 666 1044 1 ARG3 - address which receives Julian date
: 667 1045 1
: 668 1046 1 IMPLICIT OUTPUTS:
: 669 1047 1 none
: 670 1048 1
: 671 1049 1 ROUTINE VALUE:
: 672 1050 1 none
: 673 1051 1
: 674 1052 1 SIDE EFFECTS:
: 675 1053 1 none
: 676 1054 1
: 677 1055 1 USER ERRORS:
: 678 1056 1 $$$_BADATTRIB - attribute length neq 8
: 679 1057 1
: 680 1058 1 --
: 681 1059 1
: 682 1060 2 BEGIN
: 683 1061 2
: 684 1062 2 LOCAL
: 685 1063 2 REGDATE : VECTOR [12, BYTE],
: 686 1064 2 REGDESC : VECTOR [2];
: 687 1065 2
: 688 1066 2 EXTERNAL ROUTINE
: 689 1067 2 SYSSASCTIM : ADDRESSING_MODE (ABSOLUTE),
: 690 1068 2 CONVDATE_R2J;
: 691 1069 2
: 692 1070 2 REGDESC[0] = 12;
: 693 1071 2 REGDESC[1] = REGDATE;
: 694 1072 2 SYSSASCTIM(0, REGDESC, .INPUT, 0);
: 695 1073 2 CONVDATE_R2J(REGDATE, .OUTPUT);
: 696 1074 1 END;
```

.EXTRN SYSSASCTIM

0000 00000 WBINDATE:

.WORD Save nothing

: 1025

WTATTR
V04-000

K 12
16-Sep-1984 02:33:32
14-Sep-1984 12:46:51

VAX-11 Bliss-32 V4.0-742
[MTAACP.SRC]WTATTR.B32;1

Page 22
(6)

	5E		10	C2	00002
			0C	DD	00005
04	AE	08	AE	9E	00007
			7E	D4	0000C
		04	AC	DD	0000E
		08	AE	9F	00011
			7E	D4	00014
00000000G	9F		04	FB	00016
		0C	AC	DD	0001D
		0C	AE	9F	00020
0000G	CF		02	FB	00023
			04		00028

SUBL2	#16, SP
PUSHL	#12
MOVAB	REGDATE, REGDESC+4
CLRL	-(SP)
PUSHL	INPUT
PUSHAB	REGDESC
CLRL	-(SP)
CALLS	#4, @#SYSS\$ASCTIM
PUSHL	OUTPUT
PUSHAB	REGDATE
CALLS	#2, CONVDATE_R2J
RET	

:	1070
:	1071
:	1072
:	
:	
:	
:	
:	1073
:	
:	
:	1074

; Routine Size: 41 bytes, Routine Base: \$CODE\$ + 03AC


```

: 698      1075 1 ROUTINE WANSINAME ( POINTER, LENGTH ) : COMMON_CALL NOVALUE =
: 699      1076 1
: 700      1077 1 |++
: 701      1078 1
: 702      1079 1 | FUNCTIONAL DESCRIPTION:
: 703      1080 1 |     Writes the 17 or less character filename containing ANSI 'a' characters
: 704      1081 1 |     into the HDR1. The remainder of a VMS long filename is written into
: 705      1082 1 |     the HDR4 label.
: 706      1083 1 |
: 707      1084 1 | CALLING SEQUENCE:
: 708      1085 1 |     WANSINAME ( pointer, length )
: 709      1086 1 |
: 710      1087 1 | INPUT PARAMETERS:
: 711      1088 1 |     pointer - points to file name string
: 712      1089 1 |     length  - length of the file name string
: 713      1090 1 |
: 714      1091 1 | IMPLICIT INPUTS:
: 715      1092 1 |     none
: 716      1093 1 |
: 717      1094 1 | OUTPUT PARAMETERS:
: 718      1095 1 |     none
: 719      1096 1 |
: 720      1097 1 | IMPLICIT OUTPUTS:
: 721      1098 1 |     file name is stuffed into the header
: 722      1099 1 |
: 723      1100 1 | ROUTINE VALUE:
: 724      1101 1 |     none
: 725      1102 1 |
: 726      1103 1 | SIDE EFFECTS:
: 727      1104 1 |     none
: 728      1105 1 |
: 729      1106 1 | USER ERRORS:
: 730      1107 1 |
: 731      1108 1 | --
: 732      1109 1 |
: 733      1110 2 | BEGIN
: 734      1111 2 |
: 735      1112 2 | EXTERNAL REGISTER
: 736      1113 2 |     COMMON_REG;
: 737      1114 2 |
: 738      1115 2 | EXTERNAL ANSI_A_BAD      : VECTOR [ , BYTE ],
: 739      1116 2 |     ESC_CHAR             : BYTE;
: 740      1117 2 |
: 741      1118 2 | LOCAL
: 742      1119 2 |     TEMP_ID              : VECTOR [ FILE_SPEC_MAX,BYTE ],
: 743      1120 2 |     RETURN_LENGTH;
: 744      1121 2 |
: 745      1122 2 | ! check length of file name
: 746      1123 2 |
: 747      1124 2 | IF .LENGTH GTRU FILE_SPEC_MAX THEN ERR_EXIT ( SSS_BADFILENAME );
: 748      1125 2 |
: 749      1126 2 | ! Space fill the temporary storage for the FILE_ID
: 750      1127 2 |
: 751      1128 2 | CH$FILL(' ',FILE_SPEC_MAX,TEMP_ID);
: 752      1129 2 |
: 753      1130 2 | ! translate into upper case and check for non-ANSI 'a' characters
: 754      1131 2
```

```

755      1132 2 IF 0 NEQ MOVTUC ( LENGTH, .POINTER, ESC_CHAR, ANSI_A_BAD,
756      1133 2 %REF ( FILE_SPEC_MAX ), TEMP_ID)
757      1134 2 THEN ERR_EXIT ( $$$_BADFILENAME );
758      1135 2
759      1136 2 ! Move the filename into the headers and padd space left with spaces
760      1137 2
761      1138 2 CH$MOVE (HD1$$_FILEID, TEMP_ID, HDR1[HD1$T_FILEID]);
762      1139 2 CH$MOVE (HD4$$_FILEID_EXT, TEMP_ID[HD1$$_FILEID], HDR4[HD4$T_FILEID_EXT]);
763      1140 2
764      1141 2 ! Check the length of the file name. If the file name will fit in
765      1142 2 the HDR1 FILE ID then set the HDR4 length to zero. Else set up
766      1143 2 the lengths such that the HDR1 FILE ID is filed with the name
767      1144 2 then the remainder of the name is put in the HDR4 label with the
768      1145 2 size that is in the HDR4 label only.
769      1146 2 PLEASE NOTE that the actual implementation of this is different for
770      1147 2 volumes with a 4 in the VOL1 standard field as opposed to a 3 or less.
771      1148 2 This is because the new standard allows us to write any kind
772      1149 2 of data in implementation dependant fields. The old standard did not allow
773      1150 2 us to do this.
774      1151 2
775      1152 2 BEGIN
776      1153 2 BIND
777      1154 2 CVT2 = DESCRIPTOR ('!2ZW');
778      1155 2 LOCAL
779      1156 2 DESCR : VECTOR [2],
780      1157 2 MVL : REF BBLOCK;
781      1158 2 MVL = .CURRENT VCB[VCB$L_MVL];
782      1159 2 IF .LENGTH LEQU HD1$$_FILEID
783      1160 2 THEN
784      1161 2 BEGIN
785      1162 2 IF .MVL[MVL$B_STDVER] GTR 3
786      1163 2 THEN
787      1164 2 HDR4[HD4$B_FILEID_EXT_SIZE] = 0
788      1165 2 ELSE
789      1166 2 CH$FILL(0,HD4$$_FILEID_EXT_V3,HDR4[HD4$T_FILEID_EXT_V3]);
790      1167 2 END
791      1168 2 ELSE
792      1169 2 BEGIN
793      1170 2 IF .MVL[MVL$B_STDVER] GTR 3
794      1171 2 THEN
795      1172 2 HDR4[HD4$B_FILEID_EXT_SIZE] = .LENGTH - HD1$$_FILEID
796      1173 2 ELSE
797      1174 2 BEGIN
798      1175 2 LOCAL LEN;
799      1176 2 LEN = .LENGTH - HD1$$_FILEID;
800      1177 2 DESCR[0] = HD4$$_FILEID_EXT_V3;
801      1178 2 DESCR[1] = HDR4[HD4$T_FILEID_EXT_V3];
802      1179 2 $FAO(CVT2,0,DESCR,.LEN);
803      1180 2 END;
804      1181 2 END
805      1182 2 END;
806      1183 2
807      1184 2 ! strip trailing blanks from the string
808      1185 2
809      1186 2 RETURN_LENGTH = 0;
810      1187 2 DECR I FROM (.LENGTH - 1) TO 0 DO
811      1188 2 IF TEMP_ID [ .I ] NEQ ' '

```



```

: 812      1189      2
: 813      1190
: 814      1191      THEN
: 815      1192      BEGIN
: 816      1193      RETURN LENGTH = .I + 1;
: 817      1194      EXITLOOP
: 818      1195      END;
: 819      1196      ! tell the other routines the size of this name
: 820      1197      ANSI_NAME_SIZE = .RETURN_LENGTH;
: 821      1198
: 822      1199      END;      ! end of Routine
```

```

57 5A 32 21 003D5 P.AAJ: .ASCII \!2ZW\
          003D9      .BLKB 3
          00000004 003DC P.AAI: .LONG 4
          00000000 003E0      .ADDRESS P.AAJ
```

```

          CVT2=
          P.AAI
          .EXTRN ANSI_A_BAD, ESC_CHAR
```

007C 00000 WANSINAME:

				5E	A8	AE	9E	00002	.WORD	Save R2,R3,R4,R5,R6	: 1075
		00000004F		8F	08	AC	D1	00006	MOVAB	-88(SP), SP	: 1124
						04	1B	0000E	CMPL	LENGTH, #79	
			0818		8F	BF	00010		BLEQU	1\$	
004F	8F			6E	00	2C	00014	1\$:	CHMU	#2072	: 1128
					08	AE	0001B		MOVC5	#0, (SP), #32, #79, TEMP_ID	
0000G	CF	0000G	CF	04	08	AC	2F	0001D	MOVTUC	LENGTH, @POINTER, ESC_CHAR, ANSI_A_BAD, -	: 1132
				08	004F	8F	00028			#79, TEMP_ID	
						02	1D	0002D	BVS	2\$	
						51	D4	0002F	CLRL	R1	
						51	D5	00031	TSTL	R1	
						04	13	00033	BEQL	3\$	
			0818		8F	BF	00035		CHMU	#2072	: 1134
			0000G		CF	D0	00039	3\$:	MOVL	HDR1, R0	: 1138
	04	A0	08	50	11	28	0003E		MOVC3	#17, TEMP_ID, 4(R0)	
				56	0000G	CF	D0	00044	MOVL	HDR4, R6	: 1139
	05	A6	19	56	3E	28	00049		MOVC3	#62, TEMP_ID+17, 5(R6)	
				50	34	AB	D0	0004F	MOVL	52(CURRENT_VCB), MVL	: 1158
				11	08	AC	D1	00053	CMPL	LENGTH, #17	: 1159
						10	1A	00057	BGTRU	5\$	
			03		22	A0	91	00059	CMPB	34(MVL), #3	: 1162
						05	1B	0005D	BLEQU	4\$	
					04	A6	94	0005F	CLRB	4(R6)	: 1164
						33	11	00062	BRB	7\$	
			43		A6	B4	00064	4\$:	CLRW	67(R6)	: 1166
					2E	11	00067		BRB	7\$	: 1159
	51		08	AC	11	C3	00069	5\$:	SUBL3	#17, LENGTH, R1	: 1172
				03	22	A0	91	0006E	CMPB	34(MVL), #3	: 1170
						06	1B	00072	BLEQU	6\$	
			04	A6	51	90	00074		MOVB	R1, 4(R6)	: 1172
					1D	11	00078		BRB	7\$	
				50	51	D0	0007A	6\$:	MOVL	R1, LEN	: 1176
				6E	02	D0	0007D		MOVL	#2, DESCR	: 1177

WTATTR
V04-000

B 13
16-Sep-1984 02:33:32
14-Sep-1984 12:46:51

VAX-11 Bliss-32 V4.0-742
[MTAACP.SRC]WTATTR.B32;1

Page 26
(7)

04	AE	43	A6	9E	00080	MOVAB	67(R6), DESCR+4	:	1178
			50	DD	00085	PUSHL	LEN	:	1179
		04	AE	9F	00087	PUSHAB	DESCR	:	
			7E	D4	0008A	CLRL	-(SP)	:	
00000000G	00	FF68	CF	9F	0008C	PUSHAB	CVT2	:	
			04	FB	00090	CALLS	#4, SYSSFA0	:	
			51	D4	00097	CLRL	RETURN_LENGTH	:	1186
	50	08	AC	D0	00099	MOVL	LENGTH, I	:	1187
			10	11	0009D	BRB	9\$	:	
	52	08	AE40	9E	0009F	MOVAB	TEMP_ID[I], R2	:	1188
	20		52	D1	000A4	CMPL	R2, #32	:	
			06	13	000A7	BEQL	9\$	:	
	51	01	A0	9E	000A9	MOVAB	1(R0), RETURN_LENGTH	:	1191
			03	11	000AD	BRB	10\$	:	1190
	ED		50	F4	000AF	SOBGEQ	I, 8\$	:	1188
0000'	CF		51	90	000B2	MOVB	RETURN_LENGTH, ANSI_NAME_SIZE	:	1197
			04	000B7	RET			:	1199

; Routine Size: 184 bytes, Routine Base: \$CODE\$ + 03E4


```

824 1200 1 :++++
825 1201 1 :ROUTINE WLABELAST (ATTRIBUTE) : NOVALUE =
826 1202 1 :
827 1203 1 :++
828 1204 1 :
829 1205 1 :FUNCTIONAL DESCRIPTION:
830 1206 1 :   This routine validates and stores the user's supplied AST control block
831 1207 1 :
832 1208 1 :CALLING SEQUENCE:
833 1209 1 :   WLABELAST(ARG1)
834 1210 1 :
835 1211 1 :INPUT PARAMETERS:
836 1212 1 :   ARG1      - address of attribute
837 1213 1 :
838 1214 1 :IMPLICIT INPUTS:
839 1215 1 :   none
840 1216 1 :
841 1217 1 :OUTPUT PARAMETERS:
842 1218 1 :   none
843 1219 1 :
844 1220 1 :IMPLICIT OUTPUTS:
845 1221 1 :   none
846 1222 1 :
847 1223 1 :ROUTINE VALUE:
848 1224 1 :   none
849 1225 1 :
850 1226 1 :SIDE EFFECTS:
851 1227 1 :   none
852 1228 1 :
853 1229 1 :USER ERRORS:
854 1230 1 :   none
855 1231 1 :
856 1232 1 :--
857 1233 1 :
858 1234 1 :   BEGIN
859 1235 1 :
860 1236 1 :   EXTERNAL REGISTER
861 1237 1 :     COMMON_REG;
862 1238 1 :
863 1239 1 :   LOCAL
864 1240 1 :     AST          : REF BBLOCK,          ! user supplied AST control block
865 1241 1 :     FUNCTION     : BLOCK [1],          ! IO function code and modifiers
866 1242 1 :     LENGTH,
867 1243 1 :     MODE;
868 1244 1 :
869 1245 1 :   EXTERNAL
870 1246 1 :     IO_PACKET    : REF BBLOCK;          ! address of IO request packet
871 1247 1 :
872 1248 1 :   FUNCTION = .IO_PACKET[IRPSW_FUNC];
873 1249 1 :   AST = ..ATTRIBUTE; ! parameter is address of attribute
874 1250 1 :   MODE = 0;
875 1251 1 :   LENGTH = 4;
876 1252 1 :   BEGIN
877 1253 1 :
878 1254 1 :   BUILTIN
879 1255 1 :     PROBER;
880 1256 1 :
```

```
: 881      1257 1 | IF .AST NEQ 0
: 882      1258 1 |   AND
: 883      1259 1 |   ( NOT PROBER(MODE, LENGTH, .AST)
: 884      1260 1 |   OR
: 885      1261 1 |   .AST[ACB$B_TYPE] NEQ DYN$C_ACB
: 886      1262 1 |   OR
: 887      1263 1 |   NOT .FUNCTION[IO$V_ACCESS])
: 888      1264 1 | THEN
: 889      1265 1 |   ERR_EXIT(SS$_ILLBLAST);
: 890      1266 1 |
: 891      1267 1 | END;
: 892      1268 1 |
: 893      1269 1 | ! won't return any AST other than the currently recorded one
: 894      1270 1 | !
: 895      1271 1 |   KERNEL_CALL(COMPLETE_USRLBL, .AST, 0);
: 896      1272 1 | END;
: 897      1273 1 | ----
: 898      1274 1 | END
: 899      1275 1 |
: 900      1276 0 | ELUDOM
```

PSECT SUMMARY

Name	Bytes	Attributes
\$LOCKEDD1\$	9	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	1180	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	64	0	1000	00:01.9

COMMAND QUALIFIERS

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

: Size: 1113 code + 76 data bytes
: Run Time: 00:23.7
: Elapsed Time: 00:50.4
: Lines/CPU Min: 3235

WTATTR
V04-000

E 13
16-Sep-1984 02:33:32

VAX-11 Bliss-32 V4.0-742

Page 29

: Lexemes/CPU-Min: 17863
: Memory Used: 187 pages
: Compilation Complete

-S

Sy

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

MT

0256

AH-BT13A-SE
VAX/VMS V4.0

**DIGITAL
CONFIDE**

EQUIPMENT
NTIAL AND

CORPORATION
PROPRIETARY