F 7

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

```
**FILE**ID**STAINIVOL                                  H  1
```

```
  SSSSSSSS  TTTTTTTTTT    AAAAAA      IIIIII     NN      NN     IIIIII    VV      VV    OOOOOO     LL
  SSSSSSSS  TTTTTTTTTT    AAAAAA      IIIIII     NN      NN     IIIIII    VV      VV    OOOOOO     LL
SS               TT     AA      AA      II       NN      NN       II      VV      VV  OO      OO   LL
SS               TT     AA      AA      II       NN      NN       II      VV      VV  OO      OO   LL
SS               TT     AA      AA      II       NNNN    NN       II      VV      VV  OO      OO   LL
SS               TT     AA      AA      II       NNNN    NN       II      VV      VV  OO      OO   LL
  SSSSSS         TT     AA      AA      II       NN  NN  NN       II      VV      VV  OO      OO   LL
  SSSSSS         TT     AA      AA      II       NN  NN  NN       II      VV      VV  OO      OO   LL
        SS       TT     AAAAAAAAAA      II       NN    NNNN       II      VV      VV  OO      OO   LL
        SS       TT     AAAAAAAAAA      II       NN    NNNN       II      VV      VV  OO      OO   LL
        SS       TT     AA      AA      II       NN      NN       II        VV  VV    OO      OO   LL              ....
        SS       TT     AA      AA      II       NN      NN       II        VV  VV    OO      OO   LL              ....
SSSSSSSS         TT     AA      AA    IIIIII     NN      NN     IIIIII        VV        OOOOOO     LLLLLLLLLL      ....
SSSSSSSS         TT     AA      AA    IIIIII     NN      NN     IIIIII        VV        OOOOOO     LLLLLLLLLL      ....

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

```
MODULE STAINIVOL(%TITLE 'Disk volume initialization'
                IDENT = 'V04-000'
                ) =
BEGIN

!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!


!++
! FACILITY:
!       Backup/Restore
!
! ABSTRACT:
!       This module contains the routines to initialize a disk volume.
!       These routines are adapted from and must track the INIT utility.
!
! ENVIRONMENT:
!       VAX/VMS user mode.
!--
!
! AUTHOR: M. Jack, CREATION DATE: 12-Feb-1980
!
! MODIFIED BY:
!
!       V03-006 ACG0332         Andrew C. Goldstein,    29-Apr-1983  15:48
!               Add full highwater mark support
!
!       V03-005 ACG0325         Andrew C. Goldstein,     4-Apr-1983  16:04
!               Fix use of file header length symbol; add file
!               name extension.
!
!       V03-004 ACG0324         Andrew C. Goldstein,     1-Apr-1983  17:51
!               Set storage map size in VCB
!
!       V03-003 ACG0313         Andrew C. Goldstein,    12-Feb-1983  17:17
```

```
1 !                     Add routine subtitles
1 !
1 !             V03-002 MLJ48676        Martin L. Jack, 6-Sep-1982  21:04
1 !                     Set the index file EFBLK at the highest used file header.
1 !
1 !             V03-001 LMP0022         L. Mark Pilant,        5-Apr-1982  17:10
1 !                     Add support for multi-header index files.
1 !
1 !             V02-003 MLJ0081         Martin L. Jack, 26-Feb-1982  15:55
1 !                     Implement RETAINMIN and RETAINMAX for new home block fields.
1 !
1 !             V02-002 MLJ0054         Martin L. Jack, 22-Nov-1981  22:46
1 !                     Integrate GET_VM and FREE_VM jacket routines.
1 !
1 !             V02-001 ACG0211         Andrew C. Goldstein,    16-Jul-1981  10:47
1 !                     Add logic to initialize save set disks
1 !
1 !**
```

```
1 REQUIRE 'SRC$:COMMON';
1 LIBRARY 'SYS$LIBRARY:LIB';
1
1
1 LINKAGE
1         L_P$=           CALL: GLOBAL(P$=11);
1
1
1 MACRO
1         L_DECL=         EXTERNAL REGISTER P$ = 11: REF VECTOR %;
1
1
1 FORWARD ROUTINE
1         ALLOCATE:       L_P$ NOVALUE,    ! Allocate blocks
1         ALLOCATE_HOME:  L_P$ NOVALUE,    ! Allocate on home block sequence
1         INIT_ALLOCATE:  L_P$ NOVALUE,    ! Allocate file structure
1         WRITE_BLOCK:    L_P$ NOVALUE,    ! Write block by LBN
1         INIT_BITMAP:    L_P$ NOVALUE,    ! Initialize bitmap file
1         INIT_INDEX1:    L_P$ NOVALUE,    ! Initialize index file (ODS-1)
1         INIT_INDEX:     L_P$ NOVALUE,    ! Initialize index file (ODS-2)
1         INIT_MFD:       L_P$ NOVALUE,    ! Initialize MFD (ODS-2)
1         WRITE_HOMEBLOCK:L_P$ NOVALUE,    ! Write home block (ODS-2)
1         INITIALIZE_VOLUME:
1                         NOVALUE;         ! Driver for initialization
1
1
1 EXTERNAL ROUTINE
1         FREE_VM:        NOVALUE,         ! Free virtual memory
1         GET_VM,                          ! Allocate virtual memory
1         GET_ZERO_VM,                     ! Allocate and clear virtual memory
1         CHECKSUM,                        ! Compute header checksum
1         CHECKSUM2,                       ! Compute block checksum
1         CREATE_WINDOW,                   ! Create window control block
1         GET_BADBLOCKS,                   ! Get bad block data
1         MAKE_POINTER1,                   ! Make map pointer (ODS-1)
1         MAKE_POINTER,                    ! Make map pointer (ODS-2)
1         STA_ALLOC_LBN,                   ! Allocate by LBN
1         TO_ODS1_DATE:   NOVALUE;         ! Convert 64-bit time to ODS-1 format
1
1
1 EXTERNAL LITERAL
1         BACKUP$_ALLOCFAIL,
1         BACKUP$_BLKZERO,
1         BACKUP$_CLUSTER,
1         BACKUP$_LARGECNT,
1         BACKUP$_MAXBAD,
1         BACKUP$_WRITEERR;
1
1
1 G$DEFINE();                              ! Define global area
1
1
1 BUILTIN
1         ROT;
```

```
%SBTTL 'Allocation table'
LITERAL
        ALLOC_MAX=      7;                      ! Size of allocation table


MACRO
        CHANNEL=        P$[0] %,                ! Channel number
        STRUCLEV_1=     P$[1] %,                ! True if ODS-1
        HOMEBLOCK_DELTA=P$[2] %,                ! Home block delta factor
        VOLUME_SIZE=    P$[3] %,                ! Size rounded to cluster boundary
        REAL_HOMEBLOCK= P$[4] %,                ! LBN of actual secondary home block
        CLUSTER=        P$[5] %,                ! Cluster factor
        _VCB=           P$[6] %,                ! Pointer to VCB for the volume
        _DEVCHAR=       P$[7] %,                ! Pointer to device characteristics
        VCB(O,P,S,E)=           BBLOCK[._VCB,O,P,S,E] %,
        DEVCHAR(O,P,S,E)=       BBLOCK[._DEVCHAR,O,P,S,E] %,


        ! Allocation table.  Consists of 2 parallel tables, for size and LBN of
        ! allocated areas.  Each contains an entry for each piece of the disk
        ! that is allocated to something.
        !
        _ALLOC_TABLE_CNT=P$[8] %,
        _ALLOC_TABLE_LBN=P$[8+ALLOC_MAX] %,
        ALLOC_TABLE_CNT(N)=     VECTOR[_ALLOC_TABLE_CNT,N] %,
        ALLOC_TABLE_LBN(N)=     VECTOR[_ALLOC_TABLE_LBN,N] %;


LITERAL
        P$SIZE=         8 + ALLOC_MAX + ALLOC_MAX;


! Define entries of the allocation table.
!
LITERAL
        BOOTBLOCK_IDX=  0,                      ! boot block
        HOMEBLOCK1_IDX= 1,                      ! primary home block
        HOMEBLOCK2_IDX= 2,                      ! alternate home block
        IDXHDR2_IDX=    3,                      ! alternate index file header
        IDXFILE_IDX=    4,                      ! index file bitmap and file headers
        BITMAP_IDX=     5,                      ! storage bitmap
        MFD_IDX=        6;                      ! MFD


MACRO
        BOOTBLOCK_CNT=  ALLOC_TABLE_CNT[BOOTBLOCK_IDX] %,
        BOOTBLOCK_LBN=  ALLOC_TABLE_LBN[BOOTBLOCK_IDX] %,
        HOMEBLOCK1_CNT= ALLOC_TABLE_CNT[HOMEBLOCK1_IDX] %,
        HOMEBLOCK1_LBN= ALLOC_TABLE_LBN[HOMEBLOCK1_IDX] %,
        HOMEBLOCK2_CNT= ALLOC_TABLE_CNT[HOMEBLOCK2_IDX] %,
        HOMEBLOCK2_LBN= ALLOC_TABLE_LBN[HOMEBLOCK2_IDX] %,
        IDXHDR2_CNT=    ALLOC_TABLE_CNT[IDXHDR2_IDX] %,
        IDXHDR2_LBN=    ALLOC_TABLE_LBN[IDXHDR2_IDX] %,
        IDXFILE_CNT=    ALLOC_TABLE_CNT[IDXFILE_IDX] %,
        IDXFILE_LBN=    ALLOC_TABLE_LBN[IDXFILE_IDX] %,
        BITMAP_CNT=     ALLOC_TABLE_CNT[BITMAP_IDX] %,
        BITMAP_LBN=     ALLOC_TABLE_LBN[BITMAP_IDX] %,
```

```
;  189          1292  1         MFD_CNT=          ALLOC_TABLE_CNT[MFD_IDX] %,
;  190          1293  1         MFD_LBN=          ALLOC_TABLE_LBN[MFD_IDX] %;
```

```
%SBTTL 'Boot block program'
! Boot program.  The following PDP-11 program will type out the attached
! message when the volume is booted on a PDP-11, informing the user that
! this is not a system disk.
!
BIND
        BOOT_PROGRAM    = UPLIT WORD (

%O'000240',                     !    BOOTBK: NOP                   ; NOP IDENTIFIES BOOT BLOCK
%O'012706',   %O'001000',       !            MOV     #1000,SP      ; SET TEMP STACK
%O'010700',                     !            MOV     PC,R0         ; SET ADDRESS
%O'062700',   %O'000036',       !            ADD     #BOTMSG-.,R0  ; OF MESSAGE
%O'112001',                     !    10$:    MOVB    (R0)+,R1      ; GET NEXT CHARACTER
%O'001403',                     !            BEQ     20$           ; END
%O'004767',   %O'000006',       !            CALL    TYPIT         ; NO, PRINT IT
%O'000773',                     !            BR      10$           ; LOOP FOR NEXT CHARACTER
%O'000005',                     !    20$:    RESET                 ;
%O'000000',                     !            HALT                  ; HALT
                                !
                                !
%O'110137',   %O'177566',       !    TYPIT:  MOVB    R1,@#TPB      ; PRINT CHARACTER
%O'105737',   %O'177564',       !    10$:    TSTB    @#TPS         ; DONE?
%O'100375',                     !            BPL     10$           ; NO, WAIT
%O'000207'                      !            RETURN                ;
                                !
                                !
                                !    BOTMSG:

                        );

LITERAL
        BOOT_PROG_LEN   = 38;


! Boot message.  Contains the volume label.
!
BIND
        BOOT_MESSAGE    = UPLIT BYTE (13, 10, 10,
                          '                is not a system disk', 13, 10, 10, 0);


LITERAL
        BOOT_MESG_LEN   = 40;


MACRO
        BTB$T_VOLNAME   = 38, 0, 0, 0%; ! volume label in boot block message
LITERAL
        BTB$S_VOLNAME   = 12;
```

```
1 %SBTTL 'ALLOCATE - allocate space for table entry'
1 ROUTINE ALLOCATE(INDEX): L_P$ NOVALUE=
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine allocates the given table entry in the first available
1 !       position after the given start.  If none, it allocates in the first
1 !       available position before the given start.  If none, it signals an
1 !       error.
1 !
1 ! INPUT PARAMETERS:
1 !       INDEX           - Index of table entry to allocate.
1 !
1 ! IMPLICIT INPUTS:
1 !       Allocation table.
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       Allocation table.
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       NONE
1 !
1 !--
1
2 BEGIN
2 LOCAL
2       INITIAL_LBN;                    ! Initial cut at LBN
2 L_DECL;
2
2
2 ! Round the starting LBN and count down and up, respectively, to cluster
2 ! boundaries.
2 !
2 ALLOC_TABLE_LBN[.INDEX] = .ALLOC_TABLE_LBN[.INDEX] / .CLUSTER * .CLUSTER;
2 ALLOC_TABLE_CNT[.INDEX] = (.ALLOC_TABLE_CNT[.INDEX] + .CLUSTER - 1) / .CLUSTER * .CLUSTER;
2 INITIAL_LBN = .ALLOC_TABLE_LBN[.INDEX];
2
2
2 ! Iterate toward the end of the disk, checking the proposed location
2 ! of the entry.
2 !
2 WHILE TRUE DO
3     BEGIN
3
3     ! Try to allocate the entry.  If it succeeds, return.
3     !
3     IF STA_ALLOC_LBN(.ALLOC_TABLE_CNT[.INDEX], .ALLOC_TABLE_LBN[.INDEX])
3     THEN
3         RETURN;
3
```

```
3
3       ! Increment to the next cluster.
3       !
3       ALLOC_TABLE_LBN[.INDEX] = .ALLOC_TABLE_LBN[.INDEX] + .CLUSTER;
3
3
3       ! If we have fallen off the end of the volume, exit the loop.
3       !
3       IF .ALLOC_TABLE_LBN[.INDEX] GEQU .VOLUME_SIZE
3       THEN
3           EXITLOOP;
2       END;
2
2
2 ! Iterate toward the beginning of the disk, checking the proposed location
2 ! of the entry.
2 !
2 ALLOC_TABLE_LBN[.INDEX] = .INITIAL_LBN;
2 WHILE TRUE DO
3       BEGIN
3
3       ! Try to allocate the entry.  If it succeeds, return.
3       !
3       IF STA_ALLOC_LBN(.ALLOC_TABLE_CNT[.INDEX], .ALLOC_TABLE_LBN[.INDEX])
3       THEN
3           RETURN;
3
3
3       ! If we have fallen off the beginning of the volume, report failure.
3       !
3       IF .ALLOC_TABLE_LBN[.INDEX] EQL 0
3       THEN
3           SIGNAL(BACKUP$_ALLOCFAIL, 1, VCB[VCB_DEVICE]);
3
3
3       ! Decrement to the previous cluster.
3       !
3       ALLOC_TABLE_LBN[.INDEX] = .ALLOC_TABLE_LBN[.INDEX] - .CLUSTER;
2       END;
1 END;
```

```
                        .TITLE  STAINIVOL Disk volume initialization
                        .IDENT  \V04-000\

                        .PSECT  COMMON,NOEXE,  OVR,2

          00000 GLOBAL_BASE:
                        .BLKB   0
          00000 FREE_LIST:
                        .BLKB   8
          00008 INPUT_WAIT:
                        .BLKB   8
          00010 REREAD_WAIT:
                        .BLKB   8
          00018 OUTPUT_WAIT:
                        .BLKB   8
```

```
00020 JPI_UIC:.BLKB   4
00024 JPI_USERNAME:
              .BLKB   12
00030 JPI_DATE:
              .BLKB   8
00038 JPI_NODE_DESC:
              .BLKB   8
00040 JPI_CURPRIV:
              .BLKB   8
00048 SYI_VERSION:
              .BLKB   4
0004C SYI_SID:.BLKB   4
00050 RWSV_HOLD_LIST:
              .BLKB   8
00058 RWSV_CRC16:
              .BLKB   64
00098 RWSV_AUTODIN:
              .BLKB   64
000D8 RWSV_FILESET_ID:
              .BLKB   8
000E0 RWSV_VOLUME_ID:
              .BLKB   12
000EC RWSV_VOL_NUMBER:
              .BLKB   2
000EE RWSV_SEG_NUMBER:
              .BLKB   2
000F0 RWSV_FILE_NUMBER:
              .BLKB   4
000F4 RWSV_SAVE_QUAL:
              .BLKB   4
000F8 RWSV_SAVE_FAB:
              .BLKB   4
000FC RWSV_CHAN:
              .BLKB   4
00100 RWSV_XOR_BCB:
              .BLKB   4
00104 RWSV_IN_SEQ:
              .BLKB   4
00108 RWSV_IN_SEQ_0:
              .BLKB   4
0010C RWSV_IN_XOR_SEQ:
              .BLKB   4
00110 RWSV_IN_XOR_RFA:
              .BLKB   6
00116 RWSV_LOOKAHEAD:
              .BLKB   1
00117 RWSV_XORSIZE:
              .BLKB   1
00118 RWSV_IN_GROUP_SIZE:
              .BLKB   4
0011C RWSV_IN_ERRORS:
              .BLKB   2
0011E RWSV_IN_XORUSE:
              .BLKB   2
00120 RWSV_IN_ORGERR:
              .BLKB   8
00128 RWSV_IN_VBN:
```

```
                .BLKB   4
00012C RWSV_IN_VBN_0:
                .BLKB   4
00130 RWSV_ALLOC:
                .BLKB   4
00134 RWSV_EOF:
                .BLKB   4
00138 RWSV_OUT_SEQ:
                .BLKB   4
0013C RWSV_OUT_VBN:
                .BLKB   4
00140 RWSV_OUT_BLOCK_COUNT:
                .BLKB   4
00144 RWSV_OUT_ERRORS:
                .BLKB   2
00146 RWSV_SEQ_ERRORS:
                .BLKB   2
00148 RWSV_OUT_GROUP_COUNT:
                .BLKB   1
00149 RWSV_PADDING:
                .BLKB   3
0014C QUAL:     .BLKB   112
001BC COM_SSNAME:
                .BLKB   8
001C4 COM_VALID_TYPES:
                .BLKB   2
001C6 COM_FLAGS:
                .BLKB   2
001C8 COM_PADDING:
                .BLKB   1
001C9 COM_BUFF_COUNT:
                .BLKB   1
001CA COM_I_SETCOUNT:
                .BLKB   1
001CB COM_O_SETCOUNT:
                .BLKB   1
001CC COM_I_STRUCNAME:
                .BLKB   12
001D8 COM_O_STRUCNAME:
                .BLKB   12
001E4 COM_O_BSRDATE:
                .BLKB   8
001EC ALT_SSNAME:
                .BLKB   32
0020C INPUT_FUNC:
                .BLKB   1
0020D INPUT_RTYPE:
                .BLKB   1
0020E OUTPUT_FUNC:
                .BLKB   1
0020F FAST_STRUCLEV:
                .BLKB   1
00210 INPUT_BEG:
                .BLKB   0
00210 INPUT_CHAN:
                .BLKB   4
00214 INPUT_FLAGS:
```

```
                 .BLKB   2
00216 INPUT_PADDING:
                 .BLKB   2
00218 INPUT_FAB:
                 .BLKB   4
0021C INPUT_NAM:
                 .BLKB   4
00220 INPUT_BCB:
                 .BLKB   4
00224 INPUT_QUAL:
                 .BLKB   4
00228 INPUT_BAD:
                 .BLKB   4
0022C INPUT_BLOCK:
                 .BLKB   4
00230 INPUT_MAXBLOCK:
                 .BLKB   4
00234 INPUT_MEDIA_ID:
                 .BLKB   4
00238 INPUT_NAMEDESC:
                 .BLKB   8
00240 INPUT_STATBLK:
                 .BLKB   8
00248 INPUT_HDR_BEG:
                 .BLKB   0
00248 INPUT_CREDATE:
                 .BLKB   8
00250 INPUT_REVDATE:
                 .BLKB   8
00258 INPUT_EXPDATE:
                 .BLKB   8
00260 INPUT_BAKDATE:
                 .BLKB   8
00268 INPUT_FILEOWNER:
                 .BLKB   4
0026C INPUT_FILECHAR:
                 .BLKB   4
00270 INPUT_RECATTR:
                 .BLKB   32
00290 INPUT_HDR_END:
                 .BLKB   0
00290 INPUT_END:
                 .BLKB   0
00290 INPUT_PROC_LIST:
                 .BLKB   4
00294 INPUT_PLACEMENT:
                 .BLKB   8
0029C INPUT_VBN_LIST:
                 .BLKB   8
002A4 INPUT_PLACE_LEN:
                 .BLKB   2
002A6 INPUT_PADDING_2:
                 .BLKB   2
002A8 OUTPUT_BEG:
                 .BLKB   0
002A8 OUTPUT_CHAN:
                 .BLKB   4
```

```
002AC OUTPUT_FLAGS:
            .BLKB   2
002AE OUTPUT_PADDING:
            .BLKB   2
002B0 OUTPUT_FAB:
            .BLKB   4
002B4 OUTPUT_NAM:
            .BLKB   4
002B8 OUTPUT_BCB:
            .BLKB   4
002BC OUTPUT_QUAL:
            .BLKB   4
002C0 OUTPUT_BAD:
            .BLKB   4
002C4 OUTPUT_BLOCK:
            .BLKB   4
002C8 OUTPUT_MAXBLOCK:
            .BLKB   4
002CC OUTPUT_DEVGEOM:
            .BLKB   8
002D4 OUTPUT_ATTBUF:
            .BLKB   144
00364 OUTPUT_END:
            .BLKB   0
00364 LIST_TOTFILES:
            .BLKB   4
00368 LIST_TOTSIZE:
            .BLKB   4
0036C VERIFY_FAB:
            .BLKB   4
00370 VERIFY_USE_COUNT:
            .BLKB   4
00374 VERIFY_QUAL:
            .BLKB   4
00378 COMPARE_BCB:
            .BLKB   4
0037C FAST_BUFFER:
            .BLKB   4
00380 FAST_BUFFER_SIZE:
            .BLKB   4
00384 FAST_RVN:
            .BLKB   1
00385 FAST_PADDING:
            .BLKB   1
00386 DIR_VERLIMIT:
            .BLKB   2
00388 FAST_VOL_BEG:
            .BLKB   0
00388 FAST_IMAP_SIZE:
            .BLKB   4
0038C FAST_IMAP:
            .BLKB   4
00390 FAST_HDR_OFFSET:
            .BLKB   4
00394 FAST_BOOT_LBN:
            .BLKB   4
00398 FAST_VOL_END:
```

```
                        .BLKB   0
00398 JOUR_BUFFER:
                        .BLKB   4
0039C JOUR_DIR:
                        .BLKB   4
003A0 JOUR_HIBLK:
                        .BLKB   4
003A4 JOUR_EFBLK:
                        .BLKB   4
003A8 JOUR_INBLK:
                        .BLKB   4
003AC JOUR_FFBYTE:
                        .BLKB   2
003AE JOUR_INBYTE:
                        .BLKB   2
003B0 JOUR_STRUCT_LEV:
                        .BLKB   2
003B2 JOUR_COUNT:
                        .BLKB   1
003B3 JOUR_REVERSE:
                        .BLKB   1
003B4 JOUR_EXSZ:
                        .BLKB   2
003B6 JOUR_PADDING:
                        .BLKB   2
003B8 CHKPT_HIGH_SP:
                        .BLKB   4
003BC CHKPT_LOW_SP:
                        .BLKB   4
003C0 CHKPT_STACK:
                        .BLKB   4
003C4 CHKPT_VARS:
                        .BLKB   4
003C8 CHKPT_STATUS:
                        .BLKB   4
003CC DIR_BEG:.BLKB     0
003CC DIR_CHAN:
                        .BLKB   4
003D0 DIR_NAM:.BLKB     4
003D4 DIR_DEV_DESC:
                        .BLKB   4
003D8 DIR_SEL_DIR:
                        .BLKB   8
003E0 DIR_SEL_NTV:
                        .BLKB   8
003E8 DIR_STRUCLEV:
                        .BLKB   1
003E9 DIR_LEVELS:
                        .BLKB   1
003EA DIR_FLAGS:
                        .BLKB   1
003EB DIR_STATUS:
                        .BLKB   1
003EC DIR_STRING:
                        .BLKB   320
0052C DIR_STACK:
                        .BLKB   612
```

```
                                                                   00790 DIR_SP: .BLKB    4
                                                                   00794 DIR_SEL_LATEST:
                                                                                 .BLKB    4
                                                                   00798 DIR_END:.BLKB    0
                                                                   00798 DIR_SCANLIMIT:
                                                                                 .BLKB    36
                                                                   007BC INPUT_MTL:
                                                                                 .BLKB    4
                                                                   007C0 OUTPUT_MTL:
                                                                                 .BLKB    4
                                                                   007C4 CURRENT_MTL:
                                                                                 .BLKB    4
                                                                   007C8 CURRENT_VCB:
                                                                                 .BLKB    4
                                                                   007CC CURRENT_WCB:
                                                                                 .BLKB    4
                                                                   007D0 ACL_FIB_DESCR:
                                                                                 .BLKB    8
                                                                   007D8 ACL_FIB:.BLKB    64
                                                                   00818 ACL_LENGTH:
                                                                                 .BLKB    4
                                                                   0081C ACL_BUFFER:
                                                                                 .BLKB    4
                                                                   00820 CRYP_IN_CONTEXT:
                                                                                 .BLKB    4
                                                                   00824 CRYP_OU_CONTEXT:
                                                                                 .BLKB    4
                                                                   00828 CRYP_DA_CONTEXT:
                                                                                 .BLKB    4
                                                                   0082C CRYP_DATA_ENCIV:
                                                                                 .BLKB    8
                                                                   00834 CRYP_DATA_CODE:
                                                                                 .BLKB    4
                                                                   00838 CRYP_DATA_KEY:
                                                                                 .BLKB    8
                                                                   00840 CRYP_DATA_IV:
                                                                                 .BLKB    8
                                                                   00848 CRYP_DATA_CKSM:
                                                                                 .BLKB    4

                                                                                 .PSECT   CODE,NOWRT,2

0006  09F7  0303  9401  001E  65C0  11C0  0200  15C6  00A0  00000 P.AAA:    .WORD    160, 5574, 512, 4544, 26048, 30, -27647, -    ;
      0087  80FD  FF74  8BDF  FF76  905F  0000  0005  01FB  00014                    771, 2551, 6, 507, 5, 0, -28577, -138, -      ;
                                                                                     -29729, -140, -32515, 135                     ;
                                                  0A    0A    0D  00026 P.AAB:    .BYTE    13, 10, 10                                ;
73 69 20 20 20 20 20 20 20 20 20 20  20    20    20  00029           .ASCII   \                is not a system disk\        ;
64 20 6D 65 74 73 79 73 20 61 20 74  6F    6E    20  00038                                                                   ;
                                     6B    73    69  00047                                                                   ;
                                00   0A    0A    0D  0004A           .BYTE    13, 10, 10, 0                                 ;

                                                              BOOT_PROGRAM=          P.AAA
                                                              BOOT_MESSAGE=          P.AAB
                                                                         .EXTRN   FREE_VM, GET_VM
                                                                         .EXTRN   GET_ZERO_VM, CHECKSUM
                                                                         .EXTRN   CHECKSUM2, CREATE_WINDOW
                                                                         .EXTRN   GET_BADBLOCKS, MAKE_POINTER1
```

```
                                                         .EXTRN  MAKE_POINTER, STA_ALLOC_LBN
                                                         .EXTRN  TO_ODS1_DATE, BACKUP$_ALLOCFAIL
                                                         .EXTRN  BACKUP$_BLKZERO
                                                         .EXTRN  BACKUP$_CLUSTER
                                                         .EXTRN  BACKUP$_LARGECNT
                                                         .EXTRN  BACKUP$_MAXBAD, BACKUP$_WRITEERR

                                    007C 00000 ALLOCATE:
                                                         .WORD   Save R2,R3,R4,R5,R6                 ; 1344
                     56 00000000G  00   9E 00002         MOVAB   STA_ALLOC_LBN, R6
                     50        04  AC   D0 00009         MOVL    INDEX, R0                           ; 1383
                     52        3C AB40  DE 0000D         MOVAL   60(P$)[R0], R2                      ;
                     53        14  AB   D0 00012         MOVL    20(P$), R3                          ;
          51         62            53   C7 00016         DIVL3   R3, (R2), R1                        ;
          62         51            53   C5 0001A         MULL3   R3, R1, (R2)                        ;
                     54        20 AB40  DE 0001E         MOVAL   32(P$)[R0], R4                      ; 1384
          50         64            53   C1 00023         ADDL3   R3, (R4), R0                        ;
                                   50   D7 00027         DECL    R0                                  ;
                     50            53   C6 00029         DIVL2   R3, R0                              ;
          64         50            53   C5 0002C         MULL3   R3, R0, (R4)                        ;
                     55            62   D0 00030         MOVL    (R2), INITIAL_LBN                   ; 1385
                                   62   DD 00033 1$:     PUSHL   (R2)                                ; 1396
                                   64   DD 00035         PUSHL   (R4)                                ;
                     66            02   FB 00037         CALLS   #2, STA_ALLOC_LBN                   ;
                     33            50   E8 0003A         BLBS    R0, 4$                              ;
                     62            53   C0 0003D         ADDL2   R3, (R2)                            ; 1403
               0C    AB            62   D1 00040         CMPL    (R2), 12(P$)                        ; 1408
                     ED            1F   00044            BLSSU   1$                                  ;
                     62            55   D0 00046         MOVL    INITIAL_LBN, (R2)                   ; 1417
                                   62   DD 00049 2$:     PUSHL   (R2)                                ; 1423
                                   64   DD 0004B         PUSHL   (R4)                                ;
                     66            02   FB 0004D         CALLS   #2, STA_ALLOC_LBN                   ;
                     1D            50   E8 00050         BLBS    R0, 4$                              ;
                                   62   D5 00053         TSTL    (R2)                                ; 1430
                                   14   12 00055         BNEQ    3$                                  ;
          7E         18    AB      20   C1 00057         ADDL3   #32, 24(P$), -(SP)                  ; 1432
                                   01   DD 0005C         PUSHL   #1                                  ;
                        00000000G  8F   DD 0005E         PUSHL   #BACKUP$_ALLOCFAIL                  ;
           00000000G 00            03   FB 00064         CALLS   #3, LIB$SIGNAL                      ;
                     62            53   C2 0006B 3$:     SUBL2   R3, (R2)                            ; 1437
                                   D9   11 0006E         BRB     2$                                  ; 1418
                                   04      00070 4$:     RET                                         ; 1439

; Routine Size:  113 bytes,    Routine Base:  CODE + 004E
```

```
1 %SBTTL 'ALLOCATE_HOME - allocate home block'
1 ROUTINE ALLOCATE_HOME(INDEX): L_P$ NOVALUE=
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine allocates the indicated allocation table entry to
1 !       the first available block on the home block search sequence.
1 !
1 ! INPUT PARAMETERS:
1 !       INDEX           - Table index of home block cluster.
1 !
1 ! IMPLICIT INPUTS:
1 !       Allocation table.
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       Allocation table.
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       NONE
1 !
1 !--
1
2 BEGIN
2 LOCAL
2       DELTA,                                  ! home block search delta
2       BLOCKFACT,                              ! device blocking factor
2       LBN;                                    ! home block candidate LBN
2 L_DECL;
2
2 ! Compute the home block search delta.  For structure level 1, this is simply
2 ! 256, except that the first slot is on LBN 1 rather than 0.  For level 2,
2 ! compute the home block search delta from the volume geometry, according to
2 ! the following rules, where volume geometry is expressed in the order
2 ! sectors, tracks, cylinders:
2 !
2 !       n x 1 x 1:      1
2 !       1 x n x 1:      1
2 !       1 x 1 x n:      1
2 !
2 !       n x m x 1:      n+1
2 !       n x 1 x m:      n+1
2 !       1 x n x m:      n+1
2 !
2 !       s x t x c:      (t+1)*s+1
2 !
2 IF .STRUCLEV_1
2 THEN
2     DELTA = 256
2 ELSE
```

```
3      BEGIN
4      BLOCKFACT = (.DEVCHAR[DIB$B_SECTORS]
4                 * .DEVCHAR[DIB$B_TRACKS]
4                 * .DEVCHAR[DIB$W_CYLINDERS])
3                 / .DEVCHAR[DIB$L_MAXBLOCK];
3      DELTA = 1;
3      IF
3          .DEVCHAR[DIB$W_CYLINDERS] GTR 1 AND
3          .DEVCHAR[DIB$B_TRACKS] GTR 1
3      THEN
3          DELTA = .DELTA + .DEVCHAR[DIB$B_TRACKS];
3      IF
3          .DEVCHAR[DIB$B_SECTORS] GTR 1 AND
4          (.DEVCHAR[DIB$W_CYLINDERS] GTR 1 OR
4              .DEVCHAR[DIB$B_TRACKS] GTR 1)
3      THEN
3          DELTA = (.DELTA * .DEVCHAR[DIB$B_SECTORS] + .BLOCKFACT) / .BLOCKFACT;
3      IF
3          .DELTA EQL 0 OR
3          .DELTA GTRU .DEVCHAR[DIB$L_MAXBLOCK] / 10
3      THEN
3          DELTA = 1;
2      END;
2  HOMEBLOCK_DELTA = .DELTA;
2
2
2  ! Now find the first available cluster on the search sequence by starting
2  ! with LBN 1 and incrementing by the delta for each try.  If none is available,
2  ! report failure.
2  !
2  LBN = 1;
2  WHILE TRUE DO
3      BEGIN
3      ALLOC_TABLE_LBN[.INDEX] = .LBN / .CLUSTER * .CLUSTER;
3      IF STA_ALLOC_LBN(.CLUSTER, .ALLOC_TABLE_LBN[.INDEX]) THEN EXITLOOP;
3      IF .STRUCLEV_1 THEN LBN = .LBN AND NOT 1;
3      LBN = .LBN + .DELTA;
3      IF .LBN GEQU .VOLUME_SIZE
3      THEN
3          SIGNAL(BACKUP$_ALLOCFAIL, 1, VCB[VCB_DEVICE]);
2      END;
2
2
2  ! Save the LBN of the actual block.
2  !
2  REAL_HOMEBLOCK = .LBN;
1  END;
```

```
                              001C 00000 ALLOCATE_HOME:
                                              .WORD   Save R2,R3,R4           ; 1441
07        04    AB   E9 00002                 BLBC    4(P$), 1$               ; 1493
53      0100    8F   3C 00006                 MOVZWL  #256, DELTA             ; 1495
                63   11 0000B                 BRB     6$                      ;
```

```
                     50        1C   AB  D0 0000D 1$:      MOVL    28(P$), R0                          ; 1498
                     51        08   A0  9A 00011          MOVZBL  8(R0), R1                           ; 1499
                     52        09   A0  9A 00015          MOVZBL  9(R0), R2                           ;
                     51             52  C4 00019          MULL2   R2, R1                              ;
                     54        0A   A0  3C 0001C          MOVZWL  10(R0), R4                          ; 1500
                     51             54  C4 00020          MULL2   R4, R1                              ;
      52             51        70   A0  C7 00023          DIVL3   112(R0), R1, BLOCKFACT              ; 1501
                     53             01  D0 00028          MOVL    #1, DELTA                           ; 1502
                                    51  D4 0002B          CLRL    R1                                  ; 1504
                     01        0A   A0  B1 0002D          CMPW    10(R0), #1                          ;
                                    0F  1B 00031          BLEQU   2$                                  ;
                                    51  D6 00033          INCL    R1                                  ;
                     01        09   A0  91 00035          CMPB    9(R0), #1                           ; 1505
                                    07  1B 00039          BLEQU   2$                                  ;
                     54        09   A0  9A 0003B          MOVZBL  9(R0), R4                           ; 1507
                     53             54  C0 0003F          ADDL2   R4, DELTA                           ;
                     01        08   A0  91 00042 2$:      CMPB    8(R0), #1                           ; 1509
                                    17  1B 00046          BLEQU   4$                                  ;
                     06             51  E8 00048          BLBS    R1, 3$                              ; 1510
                     01        09   A0  91 0004B          CMPB    9(R0), #1                           ; 1511
                                    0E  1B 0004F          BLEQU   4$                                  ;
                     51        08   A0  9A 00051 3$:      MOVZBL  8(R0), R1                           ; 1513
                     51             53  C4 00055          MULL2   DELTA, R1                           ;
                     51             52  C0 00058          ADDL2   BLOCKFACT, R1                       ;
      53             51             52  C7 0005B          DIVL3   BLOCKFACT, R1, DELTA                ;
                                    53  D5 0005F 4$:      TSTL    DELTA                               ; 1515
                                    0A  13 00061          BEQL    5$                                  ;
      50        70   A0             0A  C7 00063          DIVL3   #10, 112(R0), R0                    ; 1516
                     50             53  D1 00068          CMPL    DELTA, R0                           ;
                                    03  1B 0006B          BLEQU   6$                                  ;
                     53             01  D0 0006D 5$:      MOVL    #1, DELTA                           ; 1518
                08   AB             53  D0 00070 6$:      MOVL    DELTA, 8(P$)                        ; 1520
                     52             01  D0 00074          MOVL    #1, LBN                             ; 1527
                     54        04   AC  D0 00077          MOVL    INDEX, R4                           ; 1530
      50             52        14   AB  C7 0007B 7$:      DIVL3   20(P$), LBN, R0                     ;
 3C AB44             50        14   AB  C5 00080          MULL3   20(P$), R0, 60(P$)[R4]              ;
                          3C AB44       DD 00087          PUSHL   60(P$)[R4]                          ; 1531
                               14   AB  DD 0008B          PUSHL   20(P$)                              ;
         00000000G   00             02  FB 0008E          CALLS   #2, STA_ALLOC_LBN                   ;
                     26             50  E8 00095          BLBS    R0, 9$                              ;
                     03        04   AB  E9 00098          BLBC    4(P$), 8$                           ; 1532
                     52             01  8A 0009C          BICB2   #1, LBN                             ;
                     52             53  C0 0009F 8$:      ADDL2   DELTA, LBN                          ; 1533
                0C   AB             52  D1 000A2          CMPL    LBN, 12(P$)                         ; 1534
                                    D3  1F 000A6          BLSSU   7$                                  ;
      7E        18   AB             20  C1 000A8          ADDL3   #32, 24(P$), -(SP)                  ; 1536
                                    01  DD 000AD          PUSHL   #1                                  ;
                          00000000G 8F  DD 000AF          PUSHL   #BACKUP$_ALLOCFAIL                  ;
         00000000G   00             03  FB 000B5          CALLS   #3, LIB$SIGNAL                      ;
                                    BD  11 000BC          BRB     7$                                  ; 1528
                10   AB             52  D0 000BE 9$:      MOVL    LBN, 16(P$)                         ; 1542
                                        04 000C2          RET                                         ; 1543

; Routine Size:  195 bytes,    Routine Base:  CODE + 00BF
```

```
1 %SBTTL 'INIT_ALLOCATE - allocate space for volume structures'
1 ROUTINE INIT_ALLOCATE: L_P$ NOVALUE=
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This is the main allocation routine.  It determines the size and
1 !       location of each portion of the file structure.  Each allocation is
1 !       done by choosing a candidate location for the section and checking
1 !       for conflicts.  If a conflict exists, a new location as near as
1 !       possible to the desired one is chosen.
1 !
1 ! INPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       NONE
1 !
1 !--
1
2 BEGIN
2 LOCAL
2       BAD:            REF BBLOCK;     ! Pointer to bad block descriptor
2 L_DECL;
2
2
2 ! Allocate the bad blocks to the bad block file.
2 !
2 BAD = OUTPUT_BAD[BAD_DESC];
2 INCR J FROM 0 TO .OUTPUT_BAD[BAD_NUMDESC]-1 DO
3     BEGIN
3     IF NOT STA_ALLOC_LBN(.BAD[BAD_COUNT], .BAD[BAD_LBN])
3     THEN
3         SIGNAL(BACKUP$_ALLOCFAIL, 1, VCB[VCB_DEVICE]);
3     BAD = .BAD + BAD_S_DESC;
3     END;
2
2
2 ! Allocate the boot block to the first available cluster (usually 0).
2 !
2 BOOTBLOCK_CNT = .CLUSTER;
2 ALLOCATE(BOOTBLOCK_IDX);
2 IF .BOOTBLOCK_LBN NEQ 0
2 THEN
2     SIGNAL(BACKUP$_BLKZERO, 1, VCB[VCB_DEVICE]);
```

```
2
2
2 ! Next allocate the primary and secondary home blocks.  If the boot block is
2 ! on LBN 0 and the cluster factor is greater than 1, then the primary home
2 ! block cluster is a dummy since the real home block is LBN 1.
2 !
2 HOMEBLOCK1_CNT = .CLUSTER;
2 IF .STRUCLEV_1
2 THEN
2     ALLOCATE_HOME(HOMEBLOCK1_IDX)
2 ELSE
3     BEGIN
3     IF .BOOTBLOCK_LBN EQL 0 AND .CLUSTER GTR 1
3     THEN
3         ALLOCATE(HOMEBLOCK1_IDX)
3     ELSE
3         ALLOCATE_HOME(HOMEBLOCK1_IDX);
3     HOMEBLOCK2_CNT = .CLUSTER;
3     ALLOCATE_HOME(HOMEBLOCK2_IDX);
2     END;
2
2
2 ! Now allocate the MFD, storage map, index file, and alternate index file
2 ! header, in that order.  This results in optimal locality of the most
2 ! frequently referenced portions of the file structure.  We do not allocate
2 ! the MFD at this time if this is an output disk being restored, since it
2 ! is allocated when it is encountered in the save set.
2 !
2 IF .VCB[VCB_SAVESET]
2 THEN
3     BEGIN
3     MFD_LBN = .OUTPUT_ATTBUF[VSR_INDEXLBN];
3     MFD_CNT = 1;
3     END;
2
2 ALLOCATE(MFD_IDX);
2 BITMAP_LBN = .OUTPUT_ATTBUF[VSR_INDEXLBN];
2 BITMAP_CNT = ((.VOLUME_SIZE/.CLUSTER + 4095) / 4096) + 1;
2 ALLOCATE(BITMAP_IDX);
2 VCB[VCB_BITMAP_LBN] = .BITMAP_LBN + 1;
2 VCB[VCB_BITMAP_SIZE] = .BITMAP_CNT - 1;
2
2
2 IDXFILE_LBN = .OUTPUT_ATTBUF[VSR_INDEXLBN];
2 IDXFILE_CNT = .OUTPUT_ATTBUF[VSR_MAXFILNUM] + (.OUTPUT_ATTBUF[VSR_MAXFILES]+4095)/4096;
2 ALLOCATE(IDXFILE_IDX);
2
2
2 IF NOT .STRUCLEV_1
2 THEN
3     BEGIN
3     IDXHDR2_CNT = .CLUSTER;
3     IDXHDR2_LBN = .IDXFILE_LBN + .HOMEBLOCK_DELTA;
3     ALLOCATE(IDXHDR2_IDX);
2     END;
1 END;
```

```
                                        00FC 00000 INIT_ALLOCATE:
                                                          .WORD   Save R2,R3,R4,R5,R6,R7                    ; 1545
                 57 00000000G   00   9E 00002            MOVAB   LIB$SIGNAL, R7                            ;
                 56     FEBF    CF   9E 00009            MOVAB   ALLOCATE, R6                              ;
                 55 00000000'   EF   9E 0000E            MOVAB   OUTPUT_ATTBUF+84, R5                      ;
 52       98     A5             08   C1 00015            ADDL3   #8, OUTPUT_BAD, BAD                       ; 1584
                 54       98    B5   D0 0001A            MOVL    @OUTPUT_BAD, R4                           ; 1585
                 53             01   CE 0001E            MNEGL   #1, J                                     ;
                                22   11 00021            BRB     3$                                        ;
                                62   DD 00023 1$:        PUSHL   (BAD)                                     ; 1587
                          04    A2   DD 00025            PUSHL   4(BAD)                                    ;
       00000000G  00            02   FB 00028            CALLS   #2, STA_ALLOC_LBN                         ;
                  10            50   E8 0002F            BLBS    R0, 2$                                    ;
 7E       18      AB            20   C1 00032            ADDL3   #32, 24(P$), -(SP)                        ; 1589
                                01   DD 00037            PUSHL   #1                                        ;
                    00000000G   8F   DD 00039            PUSHL   #BACKUP$_ALLOCFAIL                        ;
                  67            03   FB 0003F            CALLS   #3, LIB$SIGNAL                            ;
                  52            08   C0 00042 2$:        ADDL2   #8, BAD                                   ; 1590
 DA               53            54   F2 00045 3$:        AOBLSS  R4, J, 1$                                 ; 1585
          20      AB      14    AB   D0 00049            MOVL    20(P$), 32(P$)                            ; 1596
                                7E   D4 0004E            CLRL    -(SP)                                     ; 1597
                  66            01   FB 00050            CALLS   #1, ALLOCATE                              ;
                          3C    AB   D5 00053            TSTL    60(P$)                                    ; 1598
                                10   13 00056            BEQL    4$                                        ;
 7E       18      AB            20   C1 00058            ADDL3   #32, 24(P$), -(SP)                        ; 1600
                                01   DD 0005D            PUSHL   #1                                        ;
                    00000000G   8F   DD 0005F            PUSHL   #BACKUP$_BLKZERO                          ;
                  67            03   FB 00065            CALLS   #3, LIB$SIGNAL                            ;
          24      AB      14    AB   D0 00068 4$:        MOVL    20(P$), 36(P$)                            ; 1607
                  04      04    AB   E9 0006D            BLBC    4(P$), 5$                                 ; 1608
                                01   DD 00071            PUSHL   #1                                        ; 1610
                                1F   11 00073            BRB     8$                                        ;
                          3C    AB   D5 00075 5$:        TSTL    60(P$)                                    ; 1613
                                0D   12 00078            BNEQ    6$                                        ;
                  01      14    AB   D1 0007A            CMPL    20(P$), #1                                ;
                                07   15 0007E            BLEQ    6$                                        ;
                                01   DD 00080            PUSHL   #1                                        ; 1615
                  66            01   FB 00082            CALLS   #1, ALLOCATE                              ;
                                06   11 00085            BRB     7$                                        ;
                                01   DD 00087 6$:        PUSHL   #1                                        ; 1617
          71      A6            01   FB 00089            CALLS   #1, ALLOCATE_HOME                         ;
          28      AB      14    AB   D0 0008D 7$:        MOVL    20(P$), 40(P$)                            ; 1618
                                02   DD 00092            PUSHL   #2                                        ; 1619
          71      A6            01   FB 00094 8$:        CALLS   #1, ALLOCATE_HOME                         ;
                  50      18    AB   D0 00098            MOVL    24(P$), R0                                ; 1629
 08       07      A0            03   E1 0009C            BBC     #3, 7(R0), 9$                             ;
          54      AB            65   D0 000A1            MOVL    OUTPUT_ATTBUF+84, 84(P$)                  ; 1632
          38      AB            01   D0 000A5            MOVL    #1, 56(P$)                                ; 1633
                                06   DD 000A9 9$:        PUSHL   #6                                        ; 1636
                  66            01   FB 000AB            CALLS   #1, ALLOCATE                              ;
          50      AB            65   D0 000AE            MOVL    OUTPUT_ATTBUF+84, 80(P$)                  ; 1637
 50       0C      AB      14    AB   C7 000B2            DIVL3   20(P$), 12(P$), R0                        ; 1638
                  50    0FFF    C0   9E 000B8            MOVAB   4095(R0), R0                              ;
```

```
                              50 00001000    8F   C6 000BD          DIVL2   #4096, R0                                ;
                   34      AB          01    A0   9E 000C4          MOVAB   1(R0), 52(P$)                            ;
                                             05   DD 000C9          PUSHL   #5                                       ; 1639
                           66                01   FB 000CB          CALLS   #1, ALLOCATE                             ;
                           50          18    AB   D0 000CE          MOVL    24(P$), R0                               ; 1640
         0C      A0        50          AB    01   C1 000D2          ADDL3   #1, 80(P$), 12(R0)                       ;
         34      A0        34          AB    01   A3 000D8          SUBW3   #1, 52(P$), 52(R0)                       ; 1641
                           4C          AB    65   D0 000DE          MOVL    OUTPUT_ATTBUF+84, 76(P$)                 ; 1644
                 50        E0          A5 00000FFF 8F C1 000E2      ADDL3   #4095, OUTPUT_ATTBUF+52, R0              ; 1645
                              50 00001000    8F   C6 000EB          DIVL2   #4096, R0                                ;
                   30      AB          E4 B540    9E 000F2          MOVAB   @OUTPUT_ATTBUF+56[R0], 48(P$)            ;
                                             04   DD 000F8          PUSHL   #4                                       ; 1646
                           66                01   FB 000FA          CALLS   #1, ALLOCATE                             ;
                           11          04    AB   E8 000FD          BLBS    4(P$), 10$                               ; 1649
                   2C      AB          14    AB   D0 00101          MOVL    20(P$), 44(P$)                           ; 1652
         48      AB        4C          AB    08   AB C1 00106       ADDL3   8(P$), 76(P$), 72(P$)                    ; 1653
                                             03   DD 0010D          PUSHL   #3                                       ; 1654
                           66                01   FB 0010F          CALLS   #1, ALLOCATE                             ;
                                             04 00112 10$:          RET                                              ; 1656

; Routine Size:  275 bytes,     Routine Base:  CODE + 0182
```

```
  1 %SBTTL 'WRITE_BLOCK - write block to volume'
  1 ROUTINE WRITE_BLOCK(LBN,BUFFER): L_P$ NOVALUE=
  1
  1 !++
  1 !
  1 ! FUNCTIONAL DESCRIPTION:
  1 !       This routine writes a disk block by logical block number.
  1 !
  1 ! INPUT PARAMETERS:
  1 !       LBN             - Logical block number.
  1 !       BUFFER          - Pointer to buffer.
  1 !
  1 ! IMPLICIT INPUTS:
  1 !       NONE
  1 !
  1 ! OUTPUT PARAMETERS:
  1 !       NONE
  1 !
  1 ! IMPLICIT OUTPUTS:
  1 !       NONE
  1 !
  1 ! ROUTINE VALUE:
  1 !       NONE
  1 !
  1 ! SIDE EFFECTS:
  1 !       NONE
  1 !
  1 !--
  1
  2 BEGIN
  2 LOCAL
  2         STATUS,                         ! System service status
  2         IOSB:           VECTOR[4,WORD]; ! I/O status block
  2 L_DECL;
  2
  2
P 2 STATUS = $QIOW(
P 2     FUNC=IO$_WRITELBLK,
P 2     CHAN=.CHANNEL,
P 2     IOSB=IOSB,
P 2     P1=.BUFFER,
P 2     P2=512,
  2     P3=.LBN);
  2 IF .STATUS THEN STATUS = .IOSB[0];
  2 IF NOT .STATUS
  2 THEN
  2     SIGNAL(BACKUP$_WRITEERR + STS$K_SEVERE, 1, VCB[VCB_DEVICE], .STATUS);
  1 END;


                                                .EXTRN  SYS$QIOW

                              0000 00000 WRITE_BLOCK:
                                                .WORD   Save nothing                    ; 1658
            5E            08  C2 00002          SUBL2   #8, SP                          ;
                          7E  7C 00005          CLRQ    -(SP)                           ; 1699
```

```
                                     7E  D4 00007          CLRL    -(SP)                          ;
                             04      AC  DD 00009          PUSHL   LBN                            ;
                 7E        0200      8F  3C 0000C          MOVZWL  #512, -(SP)                    ;
                             08      AC  DD 00011          PUSHL   BUFFER                         ;
                                     7E  7C 00014          CLRQ    -(SP)                          ;
                             20      AE  9F 00016          PUSHAB  IOSB                           ;
                                     20  DD 00019          PUSHL   #32                            ;
                                     6B  DD 0001B          PUSHL   (P$)                           ;
                                     7E  D4 0001D          CLRL    -(SP)                          ;
           00000000G  00             0C  FB 0001F          CALLS   #12, SYS$QIOW                  ;
                      06             50  E9 00026          BLBC    STATUS, 1$                     ; 1700
                      50             6E  3C 00029          MOVZWL  IOSB, STATUS                   ;
                      16             50  E8 0002C          BLBS    STATUS, 2$                     ; 1701
                                     50  DD 0002F 1$:      PUSHL   STATUS                         ; 1703
      7E         18   AB             20  C1 00031          ADDL3   #32, 24(P$), -(SP)             ;
                                     01  DD 00036          PUSHL   #1                             ;
                           00000000G 8F  DD 00038          PUSHL   #BACKUP$_WRITEERR+4            ;
           00000000G  00             04  FB 0003E          CALLS   #4, LIB$SIGNAL                 ;
                                     04     00045 2$:      RET                                    ; 1704
```

; Routine Size: 70 bytes, Routine Base: CODE + 0295

```
1 %SBTTL 'INIT_BITMAP - initialize storage bitmap'
1 ROUTINE INIT_BITMAP: L_P$ NOVALUE=
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine initializes the contents of the volume storage bitmap.
1 !
1 ! INPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       NONE
1 !
1 !--
1
2 BEGIN
2 LOCAL
2       BUFFER:         BBLOCK[512];    ! Block buffer
2 L_DECL;
2
2
2 ! Build the storage control block and write it out.
2 !
2 CH$FILL(0, 512, BUFFER);
2 IF .STRUCLEV_1
2 THEN
3     BEGIN
3     MAP
3       BUFFER:         VECTOR;
3     LOCAL
3       BLOCK_COUNT;                    ! number of blocks in storage map
3
3     BLOCK_COUNT = .BITMAP_CNT - 1;
3     IF .BLOCK_COUNT GTRU 126 THEN BLOCK_COUNT = 0;
3     (BUFFER+3)<0,8> = .BLOCK_COUNT;
3     DECR J FROM .BLOCK_COUNT TO 1 DO BUFFER[.J] = 4096;
3     BUFFER[.BLOCK_COUNT+1] = ROT(.VOLUME_SIZE, 16);
3     END
2 ELSE
3     BEGIN
3     BUFFER[SCB$W_STRUCLEV] = SCB$C_LEVEL2 + 1;
3     BUFFER[SCB$W_CLUSTER] = .CLUSTER;
3     BUFFER[SCB$L_VOLSIZE] = .DEVCHAR[DIB$L_MAXBLOCK];
4     BUFFER[SCB$L_BLKSIZE] = (.DEVCHAR[DIB$B_SECTORS]
```

```
;  665     1762  4                                      * .DEVCHAR[DIB$B_TRACKS]
;  666     1763  4                                      * .DEVCHAR[DIB$W_CYLINDERS])
;  667     1764  3                                      / .DEVCHAR[DIB$L_MAXBLOCK];
;  668     1765  3        BUFFER[SCB$L_SECTORS] = .DEVCHAR[DIB$B_SECTORS];
;  669     1766  3        BUFFER[SCB$L_TRACKS] = .DEVCHAR[DIB$B_TRACKS];
;  670     1767  3        BUFFER[SCB$L_CYLINDER] = .DEVCHAR[DIB$W_CYLINDERS];
;  671     1768  3        CHECKSUM2(BUFFER, $BYTEOFFSET(SCB$W_CHECKSUM));
;  672     1769  2        END;
;  673     1770  2    WRITE_BLOCK(.BITMAP_LBN, BUFFER);
;  674     1771  2
;  675     1772  2    !
;  676     1773  2    ! Now initialize the bitmap to binary zeros.  This will also initialize any
;  677     1774  2    ! extra blocks that may be present due to cluster rounding.  The correct
;  678     1775  2    ! contents of the bitmap will be written at the end of the restore operation.
;  679     1776  2    !
;  680     1777  2    CH$FILL(0, 512, BUFFER);
;  681     1778  2    INCRU LBN FROM .BITMAP_LBN + 1 TO .BITMAP_LBN + .BITMAP_CNT - 1 DO
;  682     1779  2        WRITE_BLOCK(.LBN, BUFFER);
;  683     1780  1    END;



                                                  003C 00000 INIT_BITMAP:
                                                                  .WORD    Save R2,R3,R4,R5                        ; 1706
                               5E        FE00   CE   9E 00002     MOVAB    -512(SP), SP                            ;
   0200   8F         00        6E               00   2C 00007     MOVC5    #0, (SP), #0, #512, BUFFER              ; 1741
                                                6E      0000E                                                      ;
                               2C          04   AB   E9 0000F     BLBC     4(P$), 4$                               ; 1742
                     51   34   AB               01   C3 00013     SUBL3    #1, 52(P$), BLOCK_COUNT                 ; 1750
                0000007E       8F               51   D1 00018     CMPL     BLOCK_COUNT, #126                       ; 1751
                                                02   1B 0001F     BLEQU    1$                                      ;
                                                51   D4 00021     CLRL     BLOCK_COUNT                             ;
                          03   AE               51   90 00023 1$: MOVB     BLOCK_COUNT, BUFFER+3                   ; 1752
                               50          01   A1   9E 00027     MOVAB    1(R1), J                                ; 1753
                                                06   11 0002B     BRB      3$                                      ;
                        6E40             1000   8F   3C 0002D 2$: MOVZWL   #4096, BUFFER[J]                        ;
                               F7               50   F5 00033 3$: SOBGTR   J, 2$                                   ;
              04 AE41     0C   AB               10   9C 00036     ROTL     #16, 12(P$), BUFFER+4[BLOCK_COUNT]      ; 1754
                                                49   11 0003D     BRB      5$                                      ; 1742
                               6E        0201   8F   B0 0003F 4$: MOVW     #513, BUFFER                            ; 1758
                          02   AE          14   AB   B0 00044     MOVW     20(P$), BUFFER+2                        ; 1759
                               50          1C   AB   D0 00049     MOVL     28(P$), R0                              ; 1760
                          04   AE          70   A0   D0 0004D     MOVL     112(R0), BUFFER+4                       ;
                               51          08   A0   9A 00052     MOVZBL   8(R0), R1                               ; 1762
                               52          09   A0   9A 00056     MOVZBL   9(R0), R2                               ;
                               51               52   C4 0005A     MULL2    R2, R1                                  ;
                               53          0A   A0   3C 0005D     MOVZWL   10(R0), R3                              ; 1763
                               51               53   C4 00061     MULL2    R3, R1                                  ;
              08    AE         51          70   A0   C7 00064     DIVL3    112(R0), R1, BUFFER+8                   ; 1764
                          0C   AE          08   A0   9A 0006A     MOVZBL   8(R0), BUFFER+12                        ; 1765
                          10   AE          09   A0   9A 0006F     MOVZBL   9(R0), BUFFER+16                        ; 1766
                          14   AE          0A   A0   3C 00074     MOVZWL   10(R0), BUFFER+20                       ; 1767
                               7E        01FE   8F   3C 00079     MOVZWL   #510, -(SP)                             ; 1768
                                           04   AE   9F 0007E     PUSHAB   BUFFER                                  ;
                00000000G      00               02   FB 00081     CALLS    #2, CHECKSUM2                           ;
```

```
                                        5E DD 00088 5$:    PUSHL   SP                              ; 1770
                                   50   AB DD 0008A        PUSHL   80(P$)                          ;
                      FF28  CF          02 FB 0008D        CALLS   #2, WRITE_BLOCK                 ;
  0200  8F      00          6E          00 2C 00092        MOVC5   #0, (SP), #0, #512, BUFFER      ; 1777
                                        6E    00099                                                ;
                50    50    AB     34   AB C1 0009A        ADDL3   52(P$), 80(P$), R0              ; 1778
                            53     FF   A0 9E 000A0        MOVAB   -1(R0), R3                      ;
                            52     50   AB D0 000A4        MOVL    80(P$), LBN                     ;
                                        09 11 000A8        BRB     7$                              ;
                                 4004   8F BB 000AA 6$:    PUSHR   #^M<R2,SP>                      ; 1779
                      FF07  CF          02 FB 000AE        CALLS   #2, WRITE_BLOCK                 ;
                                        52 D6 000B3 7$:    INCL    LBN                             ;
                            53          52 D1 000B5        CMPL    LBN, R3                         ;
                                        F0 1B 000B8        BLEQU   6$                              ;
                                           04 000BA        RET                                     ; 1780

; Routine Size:  187 bytes,    Routine Base:  CODE + 02DB
```

```
%SBTTL 'Initial ODS-1 file header'
BIND
        INITIAL_HEADER_1 = UPLIT (
                                                        ! HEADER area
        BYTE (FH1$C_LENGTH / 2),                        ! ident area offset
        BYTE ((FH1$C_LENGTH + FI1$C_LENGTH)/2),         ! map area offset
        WORD (FID$C_INDEXF, FID$C_INDEXF),              ! file ID
        BYTE (1, 1),                                    ! structure version and level
        WORD (0),                                       ! file owner UIC
        WORD (0),                                       ! file protection
        WORD (0),                                       ! file characteristics
        BYTE (0),                                       ! record type
        BYTE (0),                                       ! record attributes
        WORD (0),                                       ! record size
        LONG (0, 0),                                    ! HIBLK and EFBLK
        WORD (0),                                       ! EOF byte offset
        BYTE (0, 0),                                    ! bucket size & VFC length
        WORD (0),                                       ! maximum record length
        WORD (0),                                       ! default extend size
        WORD (0, 0, 0, 0, 0, 0),                        ! unused record attributes

                                                        ! IDENT area
        WORD (%RAD50_11'INDEXF   SYS', 1),              ! file name, type and version
        WORD (1),                                       ! revision number
        BYTE (REP 34 OF (0)),                           ! dates

                                                        ! MAP area
        BYTE (0),                                       ! file segment number
        BYTE (0),                                       ! extension RVN
        WORD (0, 0),                                    ! extension file ID
        BYTE (1, 3),                                    ! map pointer count & LBN size
        BYTE (0),                                       ! map words in use
        BYTE ((512-2-FH1$C_LENGTH-FI1$C_LENGTH-FM1$C_LENGTH)/2));
                                                        ! map words available
```

```
%SBTTL 'INIT_INDEX1 - initialize ODS-1 index file'
ROUTINE INIT_INDEX1: L_P$ NOVALUE=

!++
!
! FUNCTIONAL DESCRIPTION:
!       This routine initializes the contents of an ODS-1 index file.
!       It writes the boot block, the home block, and the initial headers.
!
! INPUT PARAMETERS:
!       NONE
!
! IMPLICIT INPUTS:
!       NONE
!
! OUTPUT PARAMETERS:
!       NONE
!
! IMPLICIT OUTPUTS:
!       NONE
!
! ROUTINE VALUE:
!       NONE
!
! SIDE EFFECTS:
!       NONE
!
!--

BEGIN
LOCAL
        BUFFER:         BBLOCK[512],    ! Block buffer
        ALT_BUFFER:     BBLOCK[512],    ! Block buffer
        LBN,                            ! Current LBN
        BAD:            REF BBLOCK,     ! Pointer to bad block entry
        EXTENSION_FID,                  ! File number for extension headers
        IDXFILE_EXT_CNT,                ! Remaining count for the index file
        IDXFILE_EXT_LBN,                ! Starting block for the remaining count
        MAP_FULL_STATUS,                ! Map area full indicator
        UNMAPPED;                       ! Remaining count when header full
LITERAL
        IDXFILE_EXT_FID = 6;            ! first index file extension header file number
BIND
        IDENT_AREA      = BUFFER + FH1$C_LENGTH : BBLOCK,
        MAP_AREA        = BUFFER + FH1$C_LENGTH + FI1$C_LENGTH : BBLOCK,
        ALT_MAP_AREA    = ALT_BUFFER + FH1$C_LENGTH + FI1$C_LENGTH : BBLOCK;
L_DECL;


! First block to write is the boot block.  If a boot block was present on the
! input volume, write the boot program.  Otherwise, set up the message routine
! for the -11 and build the message.
!
IF .BBLOCK[OUTPUT_ATTBUF[VSR_BOOTBLOCK], DSC$W_LENGTH] NEQ 0
THEN
    BEGIN
    CH$COPY(
```

```
3               .BBLOCK[OUTPUT_ATTBUF[VSR_BOOTBLOCK], DSC$W_LENGTH],
3               .BBLOCK[OUTPUT_ATTBUF[VSR_BOOTBLOCK], DSC$A_POINTER],
3               0,
3               512, BUFFER);
3           BUFFER[4,0,32,0] = -1;          ! Initialize with invalid LBN
3           END
2       ELSE
3           BEGIN
3           CH$COPY(
3               BOOT_PROG_LEN, BOOT_PROGRAM,
3               BOOT_MESG_LEN, BOOT_MESSAGE,
3               0,
3               512, BUFFER);
3           CH$COPY(
3               .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$W_LENGTH],
3               .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$A_POINTER],
3               %C' ',
3               BTB$S_VOLNAME, BUFFER[BTB$T_VOLNAME]);
2           END;
2       WRITE_BLOCK(.BOOTBLOCK_LBN, BUFFER);
2
2
2       ! Now construct and write the home block.
2       !
2       CH$FILL(0, 512, BUFFER);
2       BUFFER[HM1$W_IBMAPSIZE] = (.OUTPUT_ATTBUF[VSR_MAXFILES] + 4095) / 4096;
2       BUFFER[HM1$L_IBMAPLBN] = ROT(.IDXFILE_LBN, 16);
2       BUFFER[HM1$W_MAXFILES] = .OUTPUT_ATTBUF[VSR_MAXFILES];
2       BUFFER[HM1$W_CLUSTER] = 1;
2       BUFFER[HM1$W_STRUCLEV] = .OUTPUT_ATTBUF[VSR_VOLSTRUCT];
2       CH$COPY(
2           .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$W_LENGTH],
2           .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$A_POINTER],
2           0,
2           HM1$S_VOLNAME, BUFFER[HM1$T_VOLNAME]);
2       (BUFFER[HM1$W_VOLOWNER])<0,8> = .(OUTPUT_ATTBUF[VSR_VOLOWNER])<0,8>;
2       (BUFFER[HM1$W_VOLOWNER])<8,8> = .(OUTPUT_ATTBUF[VSR_VOLOWNER])<16,8>;
2       BUFFER[HM1$W_PROTECT] = .OUTPUT_ATTBUF[VSR_PROTECT];
2       BUFFER[HM1$W_FILEPROT] = .OUTPUT_ATTBUF[VSR_FILEPROT];
2       BUFFER[HM1$B_WINDOW] = .OUTPUT_ATTBUF[VSR_WINDOW];
2       BUFFER[HM1$B_EXTEND] = .OUTPUT_ATTBUF[VSR_EXTEND];
2       BUFFER[HM1$B_LRU_LIM] = .OUTPUT_ATTBUF[VSR_LRU_LIM];
2       TO_ODS1_DATE(OUTPUT_ATTBUF[VSR_VOLDATE], BUFFER[HM1$T_CREDATE]);
2       BUFFER[HM1$L_SERIALNUM] = .OUTPUT_BAD[BAD_SERIAL];
2       CH$COPY(
2           .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$W_LENGTH],
2           .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$A_POINTER],
2           %C' ',
2           HM1$S_VOLNAME2, BUFFER[HM1$T_VOLNAME2]);
2       CH$COPY(
2           .BBLOCK[OUTPUT_ATTBUF[VSR_OWNERNAME], DSC$W_LENGTH],
2           .BBLOCK[OUTPUT_ATTBUF[VSR_OWNERNAME], DSC$A_POINTER],
2           %C' ',
2           HM1$S_OWNERNAME, BUFFER[HM1$T_OWNERNAME]);
2       CH$MOVE(HM1$S_FORMAT, UPLIT BYTE('DECFILE11A  '), BUFFER[HM1$T_FORMAT]);
2       CHECKSUM2(BUFFER, $BYTEOFFSET(HM1$W_CHECKSUM1));
2       CHECKSUM2(BUFFER, $BYTEOFFSET(HM1$W_CHECKSUM2));
```

```
2 WRITE_BLOCK(.HOMEBLOCK1_LBN, BUFFER);
2
2
2 ! Finish initializing the VCB, except for the index file window, which is
2 ! done after the header is written.
2 !
2 VCB[VCB_CLUSTER] = 1;
2 VCB[VCB_HDR_OFFSET] = 2 + .BUFFER[HM1$W_IBMAPSIZE];
2 VCB[VCB_MAXFILIDX] = .BUFFER[HM1$W_IBMAPSIZE] * 4096;
2 VCB[VCB_IMAP_LBN] = .IDXFILE_LBN;
2
2
2 ! Initialize the memory resident index file bitmap.  The first block contains
2 ! the initially allocated files marked in use; the rest is all zero.
2 !
2 VCB[VCB_IMAP] = GET_ZERO_VM(.BUFFER[HM1$W_IBMAPSIZE] * 512);
2 .VCB[VCB_IMAP] = %B'111';
2
2
2 ! Construct and write the index file header.
2 !
2 LBN = .IDXFILE_LBN + .BUFFER[HM1$W_IBMAPSIZE] - 1;
2 CH$COPY(
2     FH1$C_LENGTH+FI1$C_LENGTH+FM1$C_LENGTH, INITIAL_HEADER_1,
2     0,
2     512, BUFFER);
2 INCR J FROM BOOTBLOCK_IDX TO IDXFILE_IDX-1 DO
3     BEGIN
3     IF .ALLOC_TABLE_CNT[.J] NEQ 0
3     THEN MAKE_POINTER1(BUFFER, .ALLOC_TABLE_CNT[.J], .ALLOC_TABLE_LBN[.J]);
2     END;
2
2 ! Now that the basic information has been accounted for, account for the
2 ! space required to keep the index file headers.  This gets interesting
2 ! if it is necessary to generate extension headers.
2 !
2
2 IDXFILE_EXT_CNT = .ALLOC_TABLE_CNT[IDXFILE_IDX];
2 IDXFILE_EXT_LBN = .ALLOC_TABLE_LBN[IDXFILE_IDX];
2
2 IF NOT MAKE_POINTER1 (BUFFER, .IDXFILE_EXT_CNT, .IDXFILE_EXT_LBN, UNMAPPED)
2 THEN
3     BEGIN
3     EXTENSION_FID = IDXFILE_EXT_FID;
3     MAP_AREA[FM1$B_EX_RVN] = 0;
3     MAP_AREA[FM1$W_EX_FILNUM] = .EXTENSION_FID;
3     MAP_AREA[FM1$W_EX_FILSEQ] = .EXTENSION_FID;
3     MAP_AREA[FM1$B_EX_SEGNUM] = 0;
3     CHECKSUM (BUFFER);
3     WRITE_BLOCK(.LBN + FID$C_INDEXF, BUFFER);
3
3     CH$MOVE (512, BUFFER, ALT_BUFFER);
3     DO
4         BEGIN
4         ALT_BUFFER[FH1$W_FID_NUM] = .EXTENSION_FID;
4         ALT_BUFFER[FH1$W_FID_SEQ] = .EXTENSION_FID;
4         ALT_MAP_AREA[FM1$B_INUSE] = 0;
```

```
4               CH$FILL (0, 512-FH1$C_LENGTH-FI1$C_LENGTH-FM1$C_LENGTH,
4                        ALT_BUFFER+FH1$C_LENGTH+FI1$C_LENGTH+FM1$C_LENGTH);
4               IDXFILE_EXT_LBN = .IDXFILE_EXT_LBN - .UNMAPPED + .IDXFILE_EXT_CNT;
4               IDXFILE_EXT_CNT = .UNMAPPED;
4               MAP_FULL_STATUS = MAKE_POINTER1 (ALT_BUFFER, .IDXFILE_EXT_CNT, .IDXFILE_EXT_LBN, UNMAPPED);
4               ALT_MAP_AREA[FM1$W_EX_FILNUM] = .EXTENSION_FID + 1;
4               ALT_MAP_AREA[FM1$W_EX_FILSEQ] = .EXTENSION_FID + 1;
4               IF .MAP_FULL_STATUS
4               THEN
5                   BEGIN
5                   ALT_MAP_AREA[FM1$W_EX_FILNUM] = 0;
5                   ALT_MAP_AREA[FM1$W_EX_FILSEQ] = 0;
4                   END;
4               ALT_MAP_AREA[FM1$B_EX_SEGNUM] = .ALT_MAP_AREA[FM1$B_EX_SEGNUM] + 1;
4               CHECKSUM (ALT_BUFFER);
4               WRITE_BLOCK (.LBN + .EXTENSION_FID, ALT_BUFFER);
4               EXTENSION_FID = .EXTENSION_FID + 1;
4               END
3           UNTIL .MAP_FULL_STATUS;
3
3       ! Mark any created index file extension headers as in use
3       !
3
3           (.VCB[VCB_IMAP])<IDXFILE_EXT_FID-1,.EXTENSION_FID - IDXFILE_EXT_FID> = -1;
3           END
2       ELSE
3           BEGIN
3           CHECKSUM (BUFFER);
3           WRITE_BLOCK (.LBN + FID$C_INDEXF, BUFFER);
2           END;
2
2       CREATE_WINDOW(BUFFER, 1, VCB[VCB_INDEXF], 1, 0);
2
2
2       ! Construct and write the bad block file header.
2       !
2       CH$FILL(0, 512-FH1$C_LENGTH-FI1$C_LENGTH-FM1$C_LENGTH, BUFFER+FH1$C_LENGTH+FI1$C_LENGTH+FM1$C_LENGTH);
2       BUFFER[FH1$W_FID_NUM] = FID$C_BADBLK;
2       BUFFER[FH1$W_FID_SEQ] = FID$C_BADBLK;
2       MAP_AREA[FM1$W_EX_FILNUM] = 0;
2       MAP_AREA[FM1$W_EX_FILSEQ] = 0;
2       MAP_AREA[FM1$B_EX_SEGNUM] = 0;
2       MAP_AREA[FM1$B_INUSE] = 0;
2       (IDENT_AREA[FI1$W_FILENAME])<0,32> = %RAD50_11'BADBLK';
2       BAD = OUTPUT_BAD[BAD_DESC];
2       INCR J FROM 0 TO .OUTPUT_BAD[BAD_NUMDESC]-1 DO
3           BEGIN
3           IF NOT MAKE_POINTER1(BUFFER, .BAD[BAD_COUNT], .BAD[BAD_LBN])
3           THEN SIGNAL(BACKUP$_MAXBAD, 1, VCB[VCB_DEVICE]);
3           BAD = .BAD + BAD_S_DESC;
2           END;
2       CHECKSUM(BUFFER);
2       WRITE_BLOCK(.LBN + FID$C_BADBLK, BUFFER);
2
2
2       ! Construct and write the storage map file header.
2       !
```

```
;  948      2043  2 CH$FILL (0, 512-FH1$C_LENGTH-FI1$C_LENGTH-FM1$C_LENGTH, BUFFER+FH1$C_LENGTH+FI1$C_LENGTH+FM1$C_LENGTH);
;  949      2044  2 BUFFER[FH1$W_FID_NUM] = FID$C_BITMAP;
;  950      2045  2 BUFFER[FH1$W_FID_SEQ] = FID$C_BITMAP;
;  951      2046  2 MAP_AREA[FM1$B_INUSE] = 0;
;  952      2047  2 (IDENT_AREA[FI1$W_FILENAME])<0,32> = %RAD50_11'BITMAP';
;  953      2048  2 MAKE_POINTER1(BUFFER, 1, .BITMAP_LBN);
;  954      2049  2 MAKE_POINTER1(BUFFER, .BITMAP_CNT-1, .BITMAP_LBN+1);
;  955      2050  2 CHECKSUM(BUFFER);
;  956      2051  2 WRITE_BLOCK(.LBN + FID$C_BITMAP, BUFFER);
;  957      2052  1 END;


                                                      00396           .BLKB    2
                                                 17   00398 P.AAC:    .BYTE    23                                                   ;
                                                 2E   00399           .BYTE    46                                                   ;
                                       0001    0001   0039A           .WORD    1, 1                                                 ;
                                          01     01   0039E           .BYTE    1, 1                                                 ;
                                               0000   003A0           .WORD    0                                                    ;
                                               0000   003A2           .WORD    0                                                    ;
                                               0000   003A4           .WORD    0                                                    ;
                                                 00   003A6           .BYTE    0                                                    ;
                                                 00   003A7           .BYTE    0                                                    ;
                                               0000   003A8           .WORD    0                                                    ;
                                   00000000  00000000 003AA           .LONG    0, 0                                                 ;
                                               0000   003B2           .WORD    0                                                    ;
                                          00     00   003B4           .BYTE    0, 0                                                 ;
                                               0000   003B6           .WORD    0                                                    ;
                                               0000   003B8           .WORD    0                                                    ;
                   0000   0000   0000   0000   0000   0000   003BA    .WORD    0, 0, 0, 0, 0, 0                                     ;
                      7A   BB   00   00   23   06   3A   74   003C6   .RAD50   \INDEXF   SYS\                                       ;
                                               0001   003CE           .WORD    1                                                    ;
                                               0001   003D0           .WORD    1                                                    ;
                                                00#   003D2           .BYTE    0[34]                                                ;
                                                 00   003F4           .BYTE    0                                                    ;
                                                 00   003F5           .BYTE    0                                                    ;
                                       0000    0000   003F6           .WORD    0, 0                                                 ;
                                          03     01   003FA           .BYTE    1, 3                                                 ;
                                                 00   003FC           .BYTE    0                                                    ;
                                                 CC   003FD           .BYTE    -52                                                  ;
          20   20   41   31   31   45   4C   49   46   43   45   44   003FE P.AAD:   .ASCII   \DECFILE11A  \                       ;

                                                              INITIAL_HEADER_1=   P.AAC


                                              07FC 00000 INIT_INDEX1:
                                                                      .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10                      ; 1816
                              5E      FBFC    CE   9E 00002           MOVAB    -1028(SP), SP
                              50 00000000'    EF   3C 00007           MOVZWL   OUTPUT_ATTBUF+88, R0                                  ; 1868
                                              15   13 0000E           BEQL     1$                                                    ;
  0200     8F          00 00000000'    FF     50   2C 00010           MOVC5    R0, @OUTPUT_ATTBUF+92, #0, #512, BUFFER               ; 1871
                                      FE00    CD      0001B
                         FE04    CD           01   CE 0001E           MNEGL    #1, BUFFER+4                                          ; 1876
                                              22   11 00023           BRB      2$                                                    ; 1868
                FE00     CD      FBCC    CF   26   28 00025 1$:       MOVC3    #38, BOOT_PROGRAM, BUFFER                             ; 1880
  01DA     8F            00      FBEA    CF   28   2C 0002D           MOVC5    #40, BOOT_MESSAGE, #0, #474, (R3)                     ;
                                              63      00036                                                                          ;
```

```
       0C              20 00000000'  FF 00000000'  EF  2C 00037        MOVC5   OUTPUT_ATTBUF, @OUTPUT_ATTBUF+4, #32, #12, -;  1889
                                             FE26  CD     00044                BUFFER+38
                                             FE00  CD  9F 00047  2$:   PUSHAB  BUFFER                                     ;  1891
                                               3C  AB  DD 0004B        PUSHL   60(P$)                                     ;
                                FE38  CF           02  FB 0004E        CALLS   #2, WRITE_BLOCK                            ;
  0200  8F             00             6E           00  2C 00053        MOVC5   #0, (SP), #0, #512, BUFFER                 ;  1896
                                             FE00  CD     0005A                                                           ;
                       50 00000000'  EF 00000FFF  8F  C1 0005D        ADDL3   #4095, OUTPUT_ATTBUF+52, R0                ;  1897
                       51            50 00001000  8F  C7 00069        DIVL3   #4096, R0, R1                              ;
                                FE00  CD           51  B0 00071        MOVW    R1, BUFFER                                 ;
           FE02  CD              4C  AB           10  9C 00076        ROTL    #16, 76(P$), BUFFER+2                      ;  1898
                                FE06  CD 00000000'  EF  B0 0007D        MOVW    OUTPUT_ATTBUF+52, BUFFER+6                 ;  1899
                                FE08  CD           01  B0 00086        MOVW    #1, BUFFER+8                               ;  1900
                                FE0C  CD 00000000'  EF  B0 0008B        MOVW    OUTPUT_ATTBUF+64, BUFFER+12                ;  1901
       0C              00 00000000'  FF 00000000'  EF  2C 00094        MOVC5   OUTPUT_ATTBUF, @OUTPUT_ATTBUF+4, #0, #12, - ;  1906
                                             FE0E  CD     000A1                BUFFER+14                                  ;
                                FE1E  CD 00000000'  EF  90 000A4        MOVB    OUTPUT_ATTBUF+40, BUFFER+30                ;  1907
                                FE1F  CD 00000000'  EF  90 000AD        MOVB    OUTPUT_ATTBUF+42, BUFFER+31                ;  1908
                                FE20  CD 00000000'  EF  B0 000B6        MOVW    OUTPUT_ATTBUF+68, BUFFER+32                ;  1909
                                FE24  CD 00000000'  EF  B0 000BF        MOVW    OUTPUT_ATTBUF+70, BUFFER+36                ;  1910
                                FE2C  CD 00000000'  EF  90 000C8        MOVB    OUTPUT_ATTBUF+82, BUFFER+44                ;  1911
                                FE2D  CD 00000000'  EF  90 000D1        MOVB    OUTPUT_ATTBUF+76, BUFFER+45                ;  1912
                                FE2E  CD 00000000'  EF  90 000DA        MOVB    OUTPUT_ATTBUF+83, BUFFER+46                ;  1913
                                             FE3C  CD  9F 000E3        PUSHAB  BUFFER+60                                  ;  1914
                                        00000000'  EF  9F 000E7        PUSHAB  OUTPUT_ATTBUF+24                           ;
                       00000000G  00               02  FB 000ED        CALLS   #2, TO_ODS1_DATE                           ;
                                  50 00000000'  EF  D0 000F4        MOVL    OUTPUT_BAD, R0                             ;  1915
                              C8  AD          04  A0  D0 000FB        MOVL    4(R0), BUFFER+456                          ;
       0C              20 00000000'  FF 00000000'  EF  2C 00100        MOVC5   OUTPUT_ATTBUF, @OUTPUT_ATTBUF+4, #32, #12, -;  1920
                                               D8  AD     0010D                BUFFER+472                                 ;
       0C              20 00000000'  FF 00000000'  EF  2C 0010F        MOVC5   OUTPUT_ATTBUF+8, @OUTPUT_ATTBUF+12, #32, -  ;  1925
                                               E4  AD     0011C                #12, BUFFER+484                            ;
           F0  AD           FED1  CF               0C  28 0011E        MOVC3   #12, P.AAD, BUFFER+496                     ;  1926
                                                   3A  DD 00125        PUSHL   #58                                        ;  1927
                                             FE00  CD  9F 00127        PUSHAB  BUFFER                                     ;
                       00000000G  00               02  FB 0012B        CALLS   #2, CHECKSUM2                              ;
                                  7E        01FE  8F  3C 00132        MOVZWL  #510, -(SP)                                ;  1928
                                             FE00  CD  9F 00137        PUSHAB  BUFFER                                     ;
                       00000000G  00               02  FB 0013B        CALLS   #2, CHECKSUM2                              ;
                                             FE00  CD  9F 00142        PUSHAB  BUFFER                                     ;  1929
                                               40  AB  DD 00146        PUSHL   64(P$)                                     ;
                                FD3D  CF           02  FB 00149        CALLS   #2, WRITE_BLOCK                            ;
                                  52          18  AB  D0 0014E        MOVL    24(P$), R2                                 ;  1935
                              04  A2               01  B0 00152        MOVW    #1, 4(R2)                                  ;
                                  53        FE00  CD  3C 00156        MOVZWL  BUFFER, R3                                 ;  1936
           1A  A2                 53               02  A1 0015B        ADDW3   #2, R3, 26(R2)                             ;  1937
           1C  A2                 53               0C  78 00160        ASHL    #12, R3, 28(R2)                            ;
                              14  A2          4C  AB  D0 00165        MOVL    76(P$), 20(R2)                             ;  1938
                       7E         53               09  78 0016A        ASHL    #9, R3, -(SP)                              ;  1944
                       00000000G  00               01  FB 0016E        CALLS   #1, GET_ZERO_VM                            ;
                              10  A2               50  D0 00175        MOVL    R0, 16(R2)                                 ;
                              10  B2               07  D0 00179        MOVL    #7, @16(R2)                                ;  1945
                       57         53          4C  AB  C1 0017D        ADDL3   76(P$), R3, R7                             ;  1950
                                                   57  D7 00182        DECL    LBN                                        ;
  0200  8F             00   FE03  CF        0066  8F  2C 00184        MOVC5   #102, INITIAL_HEADER_1, #0, #512, BUFFER   ;  1951
                                             FE00  CD     0018F                                                           ;
                                  53          20  AB  9E 00192        MOVAB   32(P$), R3                                 ;  1957
```

```
                                         52   D4 00196        CLRL    J                                          ;
                                       6342   D5 00198 3$:    TSTL    (R3)[J]                                    ;
                                         12   13 0019B        BEQL    4$                                         ;
                                    3C AB42   DD 0019D        PUSHL   60(P$)[J]                                  ; 1958
                                       6342   DD 001A1        PUSHL   (R3)[J]                                    ;
                                  FE00   CD   9F 001A4        PUSHAB  BUFFER                                     ;
                    00000000G  00        03   FB 001A8        CALLS   #3, MAKE_POINTER1                          ;
               E5              52        03   F3 001AF 4$:    AOBLEQ  #3, J, 3$                                  ; 1955
                               59   10   A3   D0 001B3        MOVL    16(R3), IDXFILE_EXT_CNT                    ; 1966
                               58   4C   AB   D0 001B7        MOVL    76(P$), IDXFILE_EXT_LBN                    ; 1967
                               52   01   A7   9E 001BB        MOVAB   1(R7), R2                                  ; 1978
                                  4100   8F   BB 001BF        PUSHR   #^M<R8,SP>                                 ; 1969
                                         59   DD 001C3        PUSHL   IDXFILE_EXT_CNT                            ;
                                  FE00   CD   9F 001C5        PUSHAB  BUFFER                                     ;
                    00000000G  00        04   FB 001C9        CALLS   #4, MAKE_POINTER1                          ;
                               03        50   E9 001D0        BLBC    R0, 5$                                     ;
                                       00A4   31 001D3        BRW     8$                                         ;
                               56        06   D0 001D6 5$:    MOVL    #6, EXTENSION_FID                          ; 1972
                                  FE5C   CD   B4 001D9        CLRW    MAP_AREA                                   ; 1976
                         FE5E  CD        56   B0 001DD        MOVW    EXTENSION_FID, MAP_AREA+2                  ; 1974
                         FE60  CD        56   B0 001E2        MOVW    EXTENSION_FID, MAP_AREA+4                  ; 1975
                                  FE00   CD   9F 001E7        PUSHAB  BUFFER                                     ; 1977
                    00000000G  00        01   FB 001EB        CALLS   #1, CHECKSUM                               ;
                                  FE00   CD   9F 001F2        PUSHAB  BUFFER                                     ; 1978
                                         52   DD 001F6        PUSHL   R2                                         ;
                         FC8E  CF        02   FB 001F8        CALLS   #2, WRITE_BLOCK                            ;
          04   AE        FE00  CD   0200 8F   28 001FD        MOVC3   #512, BUFFER, ALT_BUFFER                   ; 1980
                           06  AE        56   B0 00206 6$:    MOVW    EXTENSION_FID, ALT_BUFFER+2                ; 1983
                           08  AE        56   B0 0020A        MOVW    EXTENSION_FID, ALT_BUFFER+4                ; 1984
                                    68   AE   94 0020E        CLRB    ALT_MAP_AREA+8                             ; 1985
 019A   8F             00      6E        00   2C 00211        MOVC5   #0, (SP), #0, #410, ALT_BUFFER+102         ; 1987
                                    6A   AE      00218                                                           ;
               50              58        6E   C3 0021A        SUBL3   UNMAPPED, IDXFILE_EXT_LBN, R0              ; 1988
               58              50        59   C1 0021E        ADDL3   IDXFILE_EXT_CNT, R0, IDXFILE_EXT_LBN       ;
                               59        6E   D0 00222        MOVL    UNMAPPED, IDXFILE_EXT_CNT                  ; 1989
                                  4100   8F   BB 00225        PUSHR   #^M<R8,SP>                                 ; 1990
                                         59   DD 00229        PUSHL   IDXFILE_EXT_CNT                            ;
                                    10   AE   9F 0022B        PUSHAB  ALT_BUFFER                                 ;
                    00000000G  00        04   FB 0022E        CALLS   #4, MAKE_POINTER1                          ;
                               5A        50   D0 00235        MOVL    R0, MAP_FULL_STATUS                        ;
                               50   01   A6   9E 00238        MOVAB   1(R6), R0                                  ; 1991
                           62  AE        50   B0 0023C        MOVW    R0, ALT_MAP_AREA+2                         ;
                           64  AE        50   B0 00240        MOVW    R0, ALT_MAP_AREA+4                         ; 1992
                               03        5A   E9 00244        BLBC    MAP_FULL_STATUS, 7$                        ; 1993
                                    62   AE   D4 00247        CLRL    ALT_MAP_AREA+2                             ; 1996
                                    60   AE   96 0024A 7$:    INCB    ALT_MAP_AREA                               ; 1999
                                    04   AE   9F 0024D        PUSHAB  ALT_BUFFER                                 ; 2000
                    00000000G  00        01   FB 00250        CALLS   #1, CHECKSUM                               ;
                                    04   AE   9F 00257        PUSHAB  ALT_BUFFER                                 ; 2001
                                       6647   9F 0025A        PUSHAB  (EXTENSION_FID)[LBN]                       ;
                         FC29  CF        02   FB 0025D        CALLS   #2, WRITE_BLOCK                            ;
                                         56   D6 00262        INCL    EXTENSION_FID                              ; 2002
                               9F        5A   E9 00264        BLBC    MAP_FULL_STATUS, 6$                        ; 2004
                               50   18   AB   D0 00267        MOVL    24(P$), R0                                 ; 2009
                               56        06   C2 0026B        SUBL2   #6, R6                                     ;
   10   B0             56      05 FFFFFFFF    8F   F0 0026E   INSV    #-1, #5, R6, @16(R0)                       ;
                                         16   11 00278        BRB     9$                                         ; 1969
```

```
                                 FE00   CD   9F 0027A 8$:     PUSHAB  BUFFER                                  ; 2013
                    00000000G  00       01   FB 0027E         CALLS   #1, CHECKSUM                            ;
                                 FE00   CD   9F 00285         PUSHAB  BUFFER                                  ; 2014
                                        52   DD 00289         PUSHL   R2                                      ;
                      FBFB  CF          02   FB 0028B         CALLS   #2, WRITE_BLOCK                         ;
                            7E          01   7D 00290 9$:     MOVQ    #1, -(SP)                               ; 2017
                                   18   AB   DD 00293         PUSHL   24(P$)                                  ;
                                        01   DD 00296         PUSHL   #1                                      ;
                                 FE00   CD   9F 00298         PUSHAB  BUFFER                                  ;
                    00000000G  00       05   FB 0029C         CALLS   #5, CREATE_WINDOW                       ;
  019A    8F           00      6E       00   2C 002A3         MOVC5   #0, (SP), #0, #410, BUFFER+102          ; 2022
                                 FE66   CD      002AA                                                         ;
                    FE02   CD 00030003  8F   D0 002AD         MOVL    #196611, BUFFER+2                       ; 2023
                                 FE5E   CD   D4 002B6         CLRL    MAP_AREA+2                              ; 2025
                                 FE5C   CD   94 002BA         CLRB    MAP_AREA                                ; 2027
                                 FE64   CD   94 002BE         CLRB    MAP_AREA+8                              ; 2028
                    FE2E   CD 0E6B0CAC  8F   D0 002C2         MOVL    #241896620, IDENT_AREA                  ; 2029
                 52 00000000'  EF       08   C1 002CB         ADDL3   #8, OUTPUT_BAD, BAD                     ; 2030
                            54 00000000'  FF D0 002D3         MOVL    @OUTPUT_BAD, R4                         ; 2031
                            53          01   CE 002DA         MNEGL   #1, J                                   ;
                                        2A   11 002DD         BRB     12$                                     ;
                                        62   DD 002DF 10$:    PUSHL   (BAD)                                   ; 2033
                                   04   A2   DD 002E1         PUSHL   4(BAD)                                  ;
                                 FE00   CD   9F 002E4         PUSHAB  BUFFER                                  ;
                    00000000G  00       03   FB 002E8         CALLS   #3, MAKE_POINTER1                       ;
                            14          50   E8 002EF         BLBS    R0, 11$                                 ;
                 7E        18  AB       20   C1 002F2         ADDL3   #32, 24(P$), -(SP)                      ; 2034
                                        01   DD 002F7         PUSHL   #1                                      ;
                             00000000G  8F   DD 002F9         PUSHL   #BACKUP$_MAXBAD                         ;
                    00000000G  00       03   FB 002FF         CALLS   #3, LIB$SIGNAL                          ;
                            52          08   C0 00306 11$:    ADDL2   #8, BAD                                 ; 2035
                 D2         53          54   F2 00309 12$:    AOBLSS  R4, J, 10$                              ; 2031
                                 FE00   CD   9F 0030D         PUSHAB  BUFFER                                  ; 2037
                    00000000G  00       01   FB 00311         CALLS   #1, CHECKSUM                            ;
                                 FE00   CD   9F 00318         PUSHAB  BUFFER                                  ; 2038
                                   03   A7   9F 0031C         PUSHAB  3(LBN)                                  ;
                      FB67  CF          02   FB 0031F         CALLS   #2, WRITE_BLOCK                         ;
  019A    8F           00      6E       00   2C 00324         MOVC5   #0, (SP), #0, #410, BUFFER+102          ; 2043
                                 FE66   CD      0032B                                                         ;
                    FE02   CD 00020002  8F   D0 0032E         MOVL    #131074, BUFFER+2                       ; 2044
                                 FE64   CD   94 00337         CLRB    MAP_AREA+8                              ; 2046
                    FE2E   CD 51780DFC  8F   D0 0033B         MOVL    #1366822396, IDENT_AREA                 ; 2047
                                   50   AB   DD 00344         PUSHL   80(P$)                                  ; 2048
                                        01   DD 00347         PUSHL   #1                                      ;
                                 FE00   CD   9F 00349         PUSHAB  BUFFER                                  ;
                    00000000G  00       03   FB 0034D         CALLS   #3, MAKE_POINTER1                       ;
                 7E        50  AB       01   C1 00354         ADDL3   #1, 80(P$), -(SP)                       ; 2049
                 7E        34  AB       01   C3 00359         SUBL3   #1, 52(P$), -(SP)                       ;
                                 FE00   CD   9F 0035E         PUSHAB  BUFFER                                  ;
                    00000000G  00       03   FB 00362         CALLS   #3, MAKE_POINTER1                       ;
                                 FE00   CD   9F 00369         PUSHAB  BUFFER                                  ; 2050
                    00000000G  00       01   FB 0036D         CALLS   #1, CHECKSUM                            ;
                                 FE00   CD   9F 00374         PUSHAB  BUFFER                                  ; 2051
                                   02   A7   9F 00378         PUSHAB  2(LBN)                                  ;
                      FB0B  CF          02   FB 0037B         CALLS   #2, WRITE_BLOCK                         ;
                                             04 00380         RET                                             ; 2052
```

```
; Routine Size:  897 bytes,    Routine Base:  CODE + 040A
```

```
%SBTTL 'Initial ODS-2 file header'
! Initial file header.  The pending bad block log file is used
! since it is the first one written. Note that this PLIT must
! be updated whenever fields are added to the file header.
!
$ASSUME (FH2$C_LENGTH, EQL, 80)
$ASSUME (FI2$C_LENGTH, EQL, 120)

BIND
        INITIAL_HEADER  = UPLIT (
                                                        ! HEADER area
        BYTE (FH2$C_LENGTH / 2),                        ! ident area offset
        BYTE ((FH2$C_LENGTH + FI2$C_LENGTH)/2),         ! map area offset
        BYTE ($BYTEOFFSET (FH2$W_CHECKSUM)/2),          ! access control list offset
        BYTE ($BYTEOFFSET (FH2$W_CHECKSUM)/2),          ! reserved area offset
        WORD (0),                                       ! file segment number
        BYTE (1, 2),                                    ! structure version and level
        WORD (FID$C_BADLOG, FID$C_BADLOG, 0),           ! file ID
        WORD (0, 0, 0),                                 ! extension file ID
        BYTE (FAT$C_FIXED),                             ! fixed length record type
        BYTE (0),                                       ! no record attributes
        WORD (16),                                      ! record size
        LONG (0, 1^16),                                 ! HIBLK and EFBLK
        WORD (0),                                       ! EOF byte offset
        BYTE (0, 0),                                    ! bucket size & VFC length
        WORD (16),                                      ! maximum record length
        WORD (0),                                       ! default extend size
        WORD (0, 0, 0, 0, 0, 0),                        ! unused record attributes
        LONG (0),                                       ! file characteristics
        WORD (0),                                       ! record protection
        BYTE (0, 0),                                    ! mapwords in use & access mode
        LONG (0),                                       ! file owner UIC
        WORD (0),                                       ! file protection
        WORD (FID$C_MFD, FID$C_MFD, 0),                 ! directory back link
        WORD (0),                                       ! journal control flags
        WORD (0),                                       ! spare
        LONG (1),                                       ! high water mark

                                                        ! IDENT area
        BYTE ('BADLOG.SYS;1          '),                ! file name, type and version
        WORD (1),                                       ! revision number
        LONG (0, 0, 0, 0, 0, 0, 0, 0),                  ! dates
        REP FI2$S_FILENAMEXT OF BYTE (' ')              ! file name extension
        );
```

```
1 %SBTTL 'INIT_INDEX - initialize ODS-2 index file'
1 ROUTINE INIT_INDEX: L_P$ NOVALUE=
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine initializes the contents of an ODS-2 index file.
1 !       It writes the boot block, the home blocks, and the initial headers.
1 !
1 ! INPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       NONE
1 !
1 !--
1
2 BEGIN
2 LOCAL
2       BUFFER:         BBLOCK[512],    ! Block buffer
2       LBN,                            ! Current LBN
2       MAP_COUNT,                      ! Count field of map pointer
2       MAP_LBN,                        ! Start LBN of current map pointer
2       BAD:            REF BBLOCK;     ! Pointer to bad block entry
2 BIND
2       IDENT_AREA      = BUFFER + FH2$C_LENGTH : BBLOCK;
2 L_DECL;
2
2
2 ! First block to write is the boot block.  If a boot block was present on the
2 ! input volume, write the boot program.  Otherwise, set up the message routine
2 ! for the -11 and build the message.
2 !
2 IF .BBLOCK[OUTPUT_ATTBUF[VSR_BOOTBLOCK], DSC$W_LENGTH] NEQ 0
2 THEN
3     BEGIN
3     CH$COPY(
3         .BBLOCK[OUTPUT_ATTBUF[VSR_BOOTBLOCK], DSC$W_LENGTH],
3         .BBLOCK[OUTPUT_ATTBUF[VSR_BOOTBLOCK], DSC$A_POINTER],
3         0,
3         512, BUFFER);
3     BUFFER[4,0,32,0] = -1;       ! Initialize with invalid LBN
3     END
2 ELSE
3     BEGIN
```

```
        CH$COPY(
            BOOT_PROG_LEN, BOOT_PROGRAM,
            BOOT_MESG_LEN, BOOT_MESSAGE,
            0,
            512, BUFFER);
        CH$COPY(
            .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$W_LENGTH],
            .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$A_POINTER],
            %C' ',
            BTB$S_VOLNAME, BUFFER[BTB$T_VOLNAME]);
        END;
    WRITE_BLOCK(.BOOTBLOCK_LBN, BUFFER);

    !
    ! Now construct the home block and write it to the remainder of the boot
    ! block cluster and to the two home block clusters.
    !
    CH$FILL(0, 512, BUFFER);
    BUFFER[HM2$L_HOMELBN] = .BOOTBLOCK_LBN + 1;
    BUFFER[HM2$L_ALHOMELBN] = .REAL_HOMEBLOCK;
    BUFFER[HM2$L_ALTIDXLBN] = .IDXHDR2_LBN;
    BUFFER[HM2$W_STRUCLEV] = .OUTPUT_ATTBUF[VSR_VOLSTRUCT];
    BUFFER[HM2$W_CLUSTER] = .CLUSTER;
    BUFFER[HM2$W_HOMEVBN] = 2;
    BUFFER[HM2$W_ALHOMEVBN] = .REAL_HOMEBLOCK - .HOMEBLOCK2_LBN + .CLUSTER * 2 + 1;
    BUFFER[HM2$W_ALTIDXVBN] = .CLUSTER * 3 + 1;
    BUFFER[HM2$W_IBMAPVBN] = .CLUSTER * 4 + 1;
    BUFFER[HM2$L_IBMAPLBN] = .IDXFILE_LBN;
    BUFFER[HM2$L_MAXFILES] = .OUTPUT_ATTBUF[VSR_MAXFILES];
    BUFFER[HM2$W_IBMAPSIZE] = (.OUTPUT_ATTBUF[VSR_MAXFILES] + 4095) / 4096;
    BUFFER[HM2$W_RESFILES] = 9;
    BUFFER[HM2$W_RVN] = .OUTPUT_ATTBUF[VSR_RVN];
    IF .OUTPUT_ATTBUF[VSR_RVN] EQL 1
    THEN
        BUFFER[HM2$W_SETCOUNT] = .COM_O_SETCOUNT;
    BUFFER[HM2$W_VOLCHAR] = .OUTPUT_ATTBUF[VSR_VOLCHAR];
    BUFFER[HM2$L_VOLOWNER] = .OUTPUT_ATTBUF[VSR_VOLOWNER];
    BUFFER[HM2$W_PROTECT] = .OUTPUT_ATTBUF[VSR_PROTECT];
    BUFFER[HM2$W_FILEPROT] = .OUTPUT_ATTBUF[VSR_FILEPROT];
    BUFFER[HM2$W_RECPROT] = .OUTPUT_ATTBUF[VSR_RECPROT];
    (BUFFER[HM2$Q_CREDATE]) = .(OUTPUT_ATTBUF[VSR_VOLDATE]);
    (BUFFER[HM2$Q_CREDATE]+4) = .(OUTPUT_ATTBUF[VSR_VOLDATE]+4);
    BUFFER[HM2$B_WINDOW] = .OUTPUT_ATTBUF[VSR_WINDOW];
    BUFFER[HM2$B_LRU_LIM] = .OUTPUT_ATTBUF[VSR_LRU_LIM];
    BUFFER[HM2$W_EXTEND] = .OUTPUT_ATTBUF[VSR_EXTEND];
    (BUFFER[HM2$Q_RETAINMIN]) = .(OUTPUT_ATTBUF[VSR_RETAINMIN]);
    (BUFFER[HM2$Q_RETAINMIN]+4) = .(OUTPUT_ATTBUF[VSR_RETAINMIN]+4);
    (BUFFER[HM2$Q_RETAINMAX]) = .(OUTPUT_ATTBUF[VSR_RETAINMAX]);
    (BUFFER[HM2$Q_RETAINMAX]+4) = .(OUTPUT_ATTBUF[VSR_RETAINMAX]+4);
    BUFFER[HM2$L_SERIALNUM] = .OUTPUT_BAD[BAD_SERIAL];
    CH$FILL(%C' ', HM2$S_STRUCNAME, BUFFER[HM2$T_STRUCNAME]);
    IF .OUTPUT_ATTBUF[VSR_RVN] NEQ 0
    THEN
        CH$MOVE(HM2$S_STRUCNAME, COM_O_STRUCNAME, BUFFER[HM2$T_STRUCNAME]);
    CH$COPY(
        .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$W_LENGTH],
        .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$A_POINTER],
```

```
	%C' ',
	HM2$S_VOLNAME, BUFFER[HM2$T_VOLNAME]);
CH$COPY(
	.BBLOCK[OUTPUT_ATTBUF[VSR_OWNERNAME], DSC$W_LENGTH],
	.BBLOCK[OUTPUT_ATTBUF[VSR_OWNERNAME], DSC$A_POINTER],
	%C' ',
	HM2$S_OWNERNAME, BUFFER[HM2$T_OWNERNAME]);
CH$MOVE(HM2$S_FORMAT, UPLIT BYTE('DECFILE11B  '), BUFFER[HM2$T_FORMAT]);
DECR J FROM .CLUSTER-1 TO 1 DO
	WRITE_HOMEBLOCK(BUFFER);
BUFFER[HM2$L_HOMELBN] = .HOMEBLOCK1_LBN;
DECR J FROM .CLUSTER TO 1 DO
	WRITE_HOMEBLOCK(BUFFER);
BUFFER[HM2$L_HOMELBN] = .HOMEBLOCK2_LBN;
DECR J FROM .CLUSTER TO 1 DO
	WRITE_HOMEBLOCK(BUFFER);


! Finish initializing the VCB, except for the index file window, which is
! done after the header is written.
!
VCB[VCB_ODS_2] = TRUE;
VCB[VCB_CLUSTER] = .BUFFER[HM2$W_CLUSTER];
VCB[VCB_HDR_OFFSET] = .BUFFER[HM2$W_CLUSTER] * 4 + .BUFFER[HM2$W_IBMAPSIZE];
VCB[VCB_MAXFILIDX] = .BUFFER[HM2$W_IBMAPSIZE] * 4096;
VCB[VCB_IMAP_LBN] = .IDXFILE_LBN;


! Initialize the memory resident index file bitmap.  The first block contains
! the initially allocated files marked in use; the rest is all zero.
!
VCB[VCB_IMAP] = GET_ZERO_VM(.BUFFER[HM2$W_IBMAPSIZE] * 512);
IF .VCB[VCB_SAVESET]
THEN .VCB[VCB_IMAP] = %B'111111111'
ELSE .VCB[VCB_IMAP] = %B'100000111';


! Construct and write the pending bad block log file header.
!
LBN = .IDXFILE_LBN + .BUFFER[HM2$W_IBMAPSIZE] - 1;
CH$COPY(
	FH2$C_LENGTH+FI2$C_LENGTH, INITIAL_HEADER,
	0,
	512, BUFFER);
BUFFER[FH2$L_FILEOWNER] = .OUTPUT_ATTBUF[VSR_VOLOWNER];
BUFFER[FH2$W_FILEPROT] = .OUTPUT_ATTBUF[VSR_FILEPROT];
BUFFER[FH2$W_RECPROT] = .OUTPUT_ATTBUF[VSR_RECPROT];
(IDENT_AREA[FI2$Q_CREDATE]) = .(OUTPUT_ATTBUF[VSR_VOLDATE]);
(IDENT_AREA[FI2$Q_CREDATE]+4) = .(OUTPUT_ATTBUF[VSR_VOLDATE]+4);
(IDENT_AREA[FI2$Q_REVDATE]) = .(OUTPUT_ATTBUF[VSR_VOLDATE]);
(IDENT_AREA[FI2$Q_REVDATE]+4) = .(OUTPUT_ATTBUF[VSR_VOLDATE]+4);
CHECKSUM(BUFFER);
WRITE_BLOCK(.LBN + FID$C_BADLOG, BUFFER);


! Construct and write the index file header.
!
```

```
2 BUFFER[FH2$W_FID_NUM] = FID$C_INDEXF;
2 BUFFER[FH2$W_FID_SEQ] = FID$C_INDEXF;
2 BBLOCK [BUFFER[FH2$W_RECATTR], FAT$W_RSIZE] = 512;
2 BBLOCK [BUFFER[FH2$W_RECATTR], FAT$W_MAXREC] = 512;
2 BBLOCK[BUFFER[FH2$W_RECATTR], FAT$L_HIBLK] =
2     ROT(.CLUSTER*4 + .IDXFILE_CNT, 16);
2 BBLOCK[BUFFER[FH2$W_RECATTR], FAT$L_EFBLK] =
2     ROT(.CLUSTER*4 + (.OUTPUT_ATTBUF[VSR_MAXFILES]+4095)/4096 + .OUTPUT_ATTBUF[VSR_MAXFILNUM] + 1, 16);
2 BUFFER[FH2$L_HIGHWATER] = .CLUSTER*4 +
2     (.OUTPUT_ATTBUF[VSR_MAXFILES]+4095)/4096 + .OUTPUT_ATTBUF[VSR_MAXFILNUM] + 1;
2 CH$MOVE(6, UPLIT BYTE('INDEXF'), IDENT_AREA[FI2$T_FILENAME]);
2 MAP_COUNT = .BOOTBLOCK_CNT;
2 MAP_LBN = .BOOTBLOCK_LBN;
2 INCR J FROM BOOTBLOCK_IDX + 1 TO IDXFILE_IDX DO
3     BEGIN
3     IF .MAP_COUNT + .MAP_LBN EQL .ALLOC_TABLE_LBN[.J]
3     THEN
3         MAP_COUNT = .MAP_COUNT + .ALLOC_TABLE_CNT[.J]
3     ELSE
4         BEGIN
4         MAKE_POINTER(BUFFER, .MAP_COUNT, .MAP_LBN);
4         MAP_COUNT = .ALLOC_TABLE_CNT[.J];
4         MAP_LBN = .ALLOC_TABLE_LBN[.J];
3         END;
2     END;
2 MAKE_POINTER(BUFFER, .MAP_COUNT, .MAP_LBN);
2 CHECKSUM(BUFFER);
2 WRITE_BLOCK(.LBN + FID$C_INDEXF, BUFFER);
2 INCR J FROM 0 TO .CLUSTER-1
2 DO WRITE_BLOCK(.IDXHDR2_LBN+.J, BUFFER);
2 CREATE_WINDOW(BUFFER, .VCB[VCB_RVN], VCB[VCB_INDEXF], 1, 0);
2 BBLOCK [.VCB[VCB_INDEXF], WCB_CUR_HWM] = .CLUSTER*4 +
2     (.OUTPUT_ATTBUF[VSR_MAXFILES]+4095)/4096 + 9 + 1;
2
2
2 ! Construct and write the bad block file header.
2 !
2 CH$FILL(0, 512-FH2$C_LENGTH-FI2$C_LENGTH, BUFFER+FH2$C_LENGTH+FI2$C_LENGTH);
2 BUFFER[FH2$B_MAP_INUSE] = 0;
2 BUFFER[FH2$W_FID_NUM] = FID$C_BADBLK;
2 BUFFER[FH2$W_FID_SEQ] = FID$C_BADBLK;
2 CH$MOVE(6, UPLIT BYTE('BADBLK'), IDENT_AREA[FI2$T_FILENAME]);
2 MAP_COUNT = 0;
2 BAD = OUTPUT_BAD[BAD_DESC];
2 INCR J FROM 0 TO .OUTPUT_BAD[BAD_NUMDESC]-1 DO
3     BEGIN
3     MAP_COUNT = .MAP_COUNT + .BAD[BAD_COUNT];
3     IF NOT MAKE_POINTER(BUFFER, .BAD[BAD_COUNT], .BAD[BAD_LBN])
3     THEN SIGNAL(BACKUP$_MAXBAD, 1, VCB[VCB_DEVICE]);
3     BAD = .BAD + BAD_S_DESC;
2     END;
2 BBLOCK[BUFFER[FH2$W_RECATTR], FAT$L_HIBLK] = ROT(.MAP_COUNT, 16);
2 BBLOCK[BUFFER[FH2$W_RECATTR], FAT$L_EFBLK] = ROT(.MAP_COUNT+1, 16);
2 BUFFER[FH2$L_HIGHWATER] = .MAP_COUNT + 1;
2 CHECKSUM(BUFFER);
2 WRITE_BLOCK(.LBN + FID$C_BADBLK, BUFFER);
2
```

```

! Construct and write the storage map file header.
!
CH$FILL(0, 512-FH2$C_LENGTH-FI2$C_LENGTH, BUFFER+FH2$C_LENGTH+FI2$C_LENGTH);
BUFFER[FH2$B_MAP_INUSE] = 0;
BUFFER[FH2$W_FID_NUM] = FID$C_BITMAP;
BUFFER[FH2$W_FID_SEQ] = FID$C_BITMAP;
BUFFER[FH2$V_CONTIG] = TRUE;
BBLOCK[BUFFER[FH2$W_RECATTR], FAT$L_HIBLK] = ROT(.BITMAP_CNT, 16);
BBLOCK[BUFFER[FH2$W_RECATTR], FAT$L_EFBLK] = ROT((.VOLUME_SIZE/.CLUSTER+4095)/4096 + 2, 16);
BUFFER[FH2$L_HIGHWATER] = (.VOLUME_SIZE/.CLUSTER+4095)/4096 + 2;
CH$MOVE(6, UPLIT BYTE('BITMAP'), IDENT_AREA[FI2$T_FILENAME]);
MAKE_POINTER(BUFFER, .BITMAP_CNT, .BITMAP_LBN);
CHECKSUM(BUFFER);
WRITE_BLOCK(.LBN + FID$C_BITMAP, BUFFER);

! If this is a save set volume we are initializing, also set up the headers
! of the miscellaneous reserved files. If the volume is being restored to,
! then the restore process will create these files.
!

IF .VCB[VCB_SAVESET]
THEN
    BEGIN

! Turn the header into the continuation file header and write it.
!

    CH$FILL (0, 512-FH2$C_LENGTH-FI2$C_LENGTH, BUFFER+FH2$C_LENGTH+FI2$C_LENGTH);
    BUFFER[FH2$B_MAP_INUSE] = 0;
    BUFFER[FH2$V_CONTIG] = FALSE;
    BBLOCK[BUFFER[FH2$W_RECATTR], FAT$L_HIBLK] = 0;
    BBLOCK[BUFFER[FH2$W_RECATTR], FAT$L_EFBLK] = 1^16;
    BUFFER[FH2$L_HIGHWATER] = 1;
    BUFFER[FH2$W_FID_NUM] = FID$C_CONTIN;
    BUFFER[FH2$W_FID_SEQ] = FID$C_CONTIN;
    CH$MOVE (6, UPLIT BYTE ('CONTIN'), IDENT_AREA[FI2$T_FILENAME]);
    CHECKSUM2 (BUFFER, $BYTEOFFSET (FH2$W_CHECKSUM));
    WRITE_BLOCK (.LBN + FID$C_CONTIN, BUFFER);

! Turn the file header into the core image file header and write it.
!

    BUFFER[FH2$W_FID_NUM] = FID$C_CORIMG;
    BUFFER[FH2$W_FID_SEQ] = FID$C_CORIMG;
    CH$MOVE (6, UPLIT BYTE ('CORIMG'), IDENT_AREA[FI2$T_FILENAME]);
    CHECKSUM2 (BUFFER, $BYTEOFFSET (FH2$W_CHECKSUM));
    WRITE_BLOCK (.LBN + FID$C_CORIMG, BUFFER);

! Turn the header into the volume set list file header and write it.
!

    BUFFER[FH2$W_FID_NUM] = FID$C_VOLSET;
    BUFFER[FH2$W_FID_SEQ] = FID$C_VOLSET;
    BBLOCK [BUFFER[FH2$W_RECATTR], FAT$W_RSIZE] = 64;
    BBLOCK [BUFFER[FH2$W_RECATTR], FAT$W_MAXREC] = 64;
    CH$MOVE (6, UPLIT BYTE ('VOLSET'), IDENT_AREA[FI2$T_FILENAME]);
```

```
3       CHECKSUM2 (BUFFER, $BYTEOFFSET (FH2$W_CHECKSUM));
3       WRITE_BLOCK (.LBN + FID$C_VOLSET, BUFFER);
3
3   ! Turn the header into the backup journal file header and write it.
3   !
3
3       BUFFER[FH2$W_FID_NUM] = FID$C_BACKUP;
3       BUFFER[FH2$W_FID_SEQ] = FID$C_BACKUP;
3       BBLOCK [BUFFER[FH2$W_RECATTR], FAT$W_RSIZE] = 512;
3       BBLOCK [BUFFER[FH2$W_RECATTR], FAT$W_MAXREC] = 512;
3       CH$MOVE (6, UPLIT BYTE ('BACKUP'), IDENT_AREA[FI2$T_FILENAME]);
3       CHECKSUM2 (BUFFER, $BYTEOFFSET (FH2$W_CHECKSUM));
3       WRITE_BLOCK (.LBN + FID$C_BACKUP, BUFFER);
3
3   ! Turn the file header into the MFD header and write it.
3   !
3
3       BUFFER[FH2$W_FID_NUM] = FID$C_MFD;
3       BUFFER[FH2$W_FID_SEQ] = FID$C_MFD;
3       BUFFER[FH2$V_CONTIG] = TRUE;
3       BUFFER[FH2$V_DIRECTORY] = 1;
3       BUFFER[FH2$L_HIGHWATER] = 2;
3       BBLOCK [BUFFER[FH2$W_RECATTR], FAT$L_EFBLK] = ROT (2, 16);
3       BBLOCK [BUFFER[FH2$W_RECATTR], FAT$L_HIBLK] = ROT (.MFD_CNT, 16);
3       BBLOCK [BUFFER[FH2$W_RECATTR], FAT$B_RTYPE] = FAT$C_VARIABLE;
3       BBLOCK [BUFFER[FH2$W_RECATTR], FAT$B_RATTRIB] = FAT$M_NOSPAN;
3
3       CH$MOVE (10, UPLIT BYTE ('000000.DIR'), IDENT_AREA[FI2$T_FILENAME]);
3       MAKE_POINTER (BUFFER, .MFD_CNT, .MFD_LBN);
3       CHECKSUM2 (BUFFER, $BYTEOFFSET (FH2$W_CHECKSUM));
3       WRITE_BLOCK (.LBN + FID$C_MFD, BUFFER);
2       END;
2
1   END;
```

```
                                          0078B          .BLKB   1
                                     28   0078C  P.AAE:  .BYTE   40
                                     64   0078D          .BYTE   100
                                     FF   0078E          .BYTE   -1
                                     FF   0078F          .BYTE   -1
                                   0000   00790          .WORD   0
                                02   01   00792          .BYTE   1, 2
                    0000   0009  0009   00794          .WORD   9, 9, 0
                    0000   0000  0000   0079A          .WORD   0, 0, 0
                                     01   007A0          .BYTE   1
                                     00   007A1          .BYTE   0
                                   0010   007A2          .WORD   16
                     00010000  00000000   007A4          .LONG   0, 65536
                                   0000   007AC          .WORD   0
                                00   00   007AE          .BYTE   0, 0
                                   0010   007B0          .WORD   16
                                   0000   007B2          .WORD   0
0000   0000   0000   0000   0000   0000   007B4          .WORD   0, 0, 0, 0, 0, 0
                               00000000   007C0          .LONG   0
                                   0000   007C4          .WORD   0
```

```
                                                       00  00  007C6          .BYTE   0, 0
                                                     00000000  007C8          .LONG   0
                                                         0000  007CC          .WORD   0
                                           0000  0004    0004  007CE          .WORD   4, 4, 0
                                                         0000  007D4          .WORD   0
                                                         0000  007D6          .WORD   0
                                                     00000001  007D8          .LONG   1
20  20  20  31  3B  53  59  53  2E  47  4F  4C  44  41  42  007DC          .ASCII  \BADLOG.SYS;1       \
                                        20  20  20  20  20  007EB
                                                         0001  007F0          .WORD   1
00000000  00000000  00000000  00000000  00000000  00000000  007F2          .LONG   0, 0, 0, 0, 0, 0, 0, 0
                                        00000000  00000000  0080A
                                                           20  00812          .ASCII  \ \
                                                           20  00813          .ASCII  \ \
                                                           20  00814          .ASCII  \ \
                                                           20  00815          .ASCII  \ \
                                                           20  00816          .ASCII  \ \
                                                           20  00817          .ASCII  \ \
                                                           20  00818          .ASCII  \ \
                                                           20  00819          .ASCII  \ \
                                                           20  0081A          .ASCII  \ \
                                                           20  0081B          .ASCII  \ \
                                                           20  0081C          .ASCII  \ \
                                                           20  0081D          .ASCII  \ \
                                                           20  0081E          .ASCII  \ \
                                                           20  0081F          .ASCII  \ \
                                                           20  00820          .ASCII  \ \
                                                           20  00821          .ASCII  \ \
                                                           20  00822          .ASCII  \ \
                                                           20  00823          .ASCII  \ \
                                                           20  00824          .ASCII  \ \
                                                           20  00825          .ASCII  \ \
                                                           20  00826          .ASCII  \ \
                                                           20  00827          .ASCII  \ \
                                                           20  00828          .ASCII  \ \
                                                           20  00829          .ASCII  \ \
                                                           20  0082A          .ASCII  \ \
                                                           20  0082B          .ASCII  \ \
                                                           20  0082C          .ASCII  \ \
                                                           20  0082D          .ASCII  \ \
                                                           20  0082E          .ASCII  \ \
                                                           20  0082F          .ASCII  \ \
                                                           20  00830          .ASCII  \ \
                                                           20  00831          .ASCII  \ \
                                                           20  00832          .ASCII  \ \
                                                           20  00833          .ASCII  \ \
                                                           20  00834          .ASCII  \ \
                                                           20  00835          .ASCII  \ \
                                                           20  00836          .ASCII  \ \
                                                           20  00837          .ASCII  \ \
                                                           20  00838          .ASCII  \ \
                                                           20  00839          .ASCII  \ \
                                                           20  0083A          .ASCII  \ \
                                                           20  0083B          .ASCII  \ \
                                                           20  0083C          .ASCII  \ \
                                                           20  0083D          .ASCII  \ \
                                                           20  0083E          .ASCII  \ \
```

```
                                              20  0083F          .ASCII  \ \                                   ;
                                              20  00840          .ASCII  \ \                                   ;
                                              20  00841          .ASCII  \ \                                   ;
                                              20  00842          .ASCII  \ \                                   ;
                                              20  00843          .ASCII  \ \                                   ;
                                              20  00844          .ASCII  \ \                                   ;
                                              20  00845          .ASCII  \ \                                   ;
                                              20  00846          .ASCII  \ \                                   ;
                                              20  00847          .ASCII  \ \                                   ;
                                              20  00848          .ASCII  \ \                                   ;
                                              20  00849          .ASCII  \ \                                   ;
                                              20  0084A          .ASCII  \ \                                   ;
                                              20  0084B          .ASCII  \ \                                   ;
                                              20  0084C          .ASCII  \ \                                   ;
                                              20  0084D          .ASCII  \ \                                   ;
                                              20  0084E          .ASCII  \ \                                   ;
                                              20  0084F          .ASCII  \ \                                   ;
                                              20  00850          .ASCII  \ \                                   ;
                                              20  00851          .ASCII  \ \                                   ;
                                              20  00852          .ASCII  \ \                                   ;
                                              20  00853          .ASCII  \ \                                   ;
      20  20  42  31  31  45  4C  49  46  43  45  44  00854 P.AAF:  .ASCII  \DECFILE11B  \                       ;
                              46  58  45  44  4E  49  00860 P.AAG:  .ASCII  \INDEXF\                           ;
                              4B  4C  42  44  41  42  00866 P.AAH:  .ASCII  \BADBLK\                           ;
                              50  41  4D  54  49  42  0086C P.AAI:  .ASCII  \BITMAP\                           ;
                              4E  49  54  4E  4F  43  00872 P.AAJ:  .ASCII  \CONTIN\                           ;
                              47  4D  49  52  4F  43  00878 P.AAK:  .ASCII  \CORIMG\                           ;
                              54  45  53  4C  4F  56  0087E P.AAL:  .ASCII  \VOLSET\                           ;
                              50  55  4B  43  41  42  00884 P.AAM:  .ASCII  \BACKUP\                           ;
              52  49  44  2E  30  30  30  30  30  30  0088A P.AAN:  .ASCII  \000000.DIR\                       ;

                                                        INITIAL_HEADER=     P.AAE


                                            07FC 00000 INIT_INDEX:
                                                                 .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10     ; 2098
                         5A       F9FB  CF  9E 00002             MOVAB   WRITE_BLOCK, R10                       ;
                         59  00000000'  EF  9E 00007             MOVAB   OUTPUT_ATTBUF+52, R9                   ;
                         5E       FE00  CE  9E 0000E             MOVAB   -512(SP), SP                           ;
                         50         24  A9  3C 00013             MOVZWL  OUTPUT_ATTBUF+88, R0                   ; 2142
                                        0F  13 00017             BEQL    1$                                     ;
  0200  8F       00       28  B9        50  2C 00019             MOVC5   R0, @OUTPUT_ATTBUF+92, #0, #512, BUFFER ; 2145
                                        6E     00021                                                            ;
                          04  AE        01  CE 00022             MNEGL   #1, BUFFER+4                           ; 2150
                                        19  11 00026             BRB     2$                                     ; 2142
                 6E     F73F  CF        26  28 00028 1$:         MOVC3   #38, BOOT_PROGRAM, BUFFER              ; 2154
  01DA  8F       00     F75F  CF        28  2C 0002E             MOVC5   #40, BOOT_MESSAGE, #0, #474, (R3)      ;
                                        63     00037                                                            ;
        0C       20       D0  B9    CC  A9  2C 00038             MOVC5   OUTPUT_ATTBUF, @OUTPUT_ATTBUF+4, #32, #12, -; 2163
                                    26  AE     0003F                     BUFFER+38                              ;
                                        5E  DD 00041 2$:         PUSHL   SP                                     ; 2165
                                    3C  AB  DD 00043             PUSHL   60(P$)                                 ;
                          6A            02  FB 00046             CALLS   #2, WRITE_BLOCK                        ;
  0200  8F       00       6E            00  2C 00049             MOVC5   #0, (SP), #0, #512, BUFFER             ; 2171
                                        6E     00050                                                            ;
                          50        3C  AB  9E 00051             MOVAB   60(P$), R0                             ; 2172
                 6E       60            01  C1 00055             ADDL3   #1, (R0), BUFFER                       ;
```

```
                    04   AE        10   AB  D0 00059        MOVL    16(P$), BUFFER+4                                ; 2173
                    08   AE        0C   A0  D0 0005E        MOVL    12(R0), BUFFER+8                                ; 2174
                    0C   AE        0C   A9  B0 00063        MOVW    OUTPUT_ATTBUF+64, BUFFER+12                     ; 2175
                         56        14   AB  D0 00068        MOVL    20(P$), R6                                      ; 2176
                    0E   AE             56  B0 0006C        MOVW    R6, BUFFER+14                                   ;
                    10   AE             02  B0 00070        MOVW    #2, BUFFER+16                                   ; 2177
             51     10   AB        08   A0  C3 00074        SUBL3   8(R0), 16(P$), R1                               ; 2178
                         52     01 A146 3E 0007A        MOVAW   1(R1)[R6], R2                                   ;
                    12   AE             52  B0 0007F        MOVW    R2, BUFFER+18                                   ;
             52          56             03  C5 00083        MULL3   #3, R6, R2                                      ; 2179
       14    AE          52             01  A1 00087        ADDW3   #1, R2, BUFFER+20                               ;
             51          56             02  78 0008C        ASHL    #2, R6, R1                                      ; 2180
       16    AE          51             01  A1 00090        ADDW3   #1, R1, BUFFER+22                               ;
                    18   AE        10   A0  D0 00095        MOVL    16(R0), BUFFER+24                               ; 2181
                    1C   AE             69  D0 0009A        MOVL    OUTPUT_ATTBUF+52, BUFFER+28                     ; 2182
             50          69 00000FFF    8F  C1 0009E        ADDL3   #4095, OUTPUT_ATTBUF+52, R0                     ; 2183
             51          50 00001000    8F  C7 000A6        DIVL3   #4096, R0, R1                                   ;
                    20   AE             51  B0 000AE        MOVW    R1, BUFFER+32                                   ;
                    22   AE             09  B0 000B2        MOVW    #9, BUFFER+34                                   ; 2184
                         57        0E   A9  3C 000B6        MOVZWL  OUTPUT_ATTBUF+66, R7                            ; 2185
                    26   AE             57  B0 000BA        MOVW    R7, BUFFER+38                                   ;
                         01             57  B1 000BE        CMPW    R7, #1                                          ; 2186
                                        06  12 000C1        BNEQ    3$                                              ;
                    28   AE      FEC3   C9  9B 000C3        MOVZBW  COM_O_SETCOUNT, BUFFER+40                       ; 2188
                    2A   AE        16   A9  B0 000C9 3$:    MOVW    OUTPUT_ATTBUF+74, BUFFER+42                     ; 2189
                    2C   AE        F4   A9  D0 000CE        MOVL    OUTPUT_ATTBUF+40, BUFFER+44                     ; 2190
                    34   AE        10   A9  D0 000D3        MOVL    OUTPUT_ATTBUF+68, BUFFER+52                     ; 2191
                    38   AE        14   A9  B0 000D8        MOVW    OUTPUT_ATTBUF+72, BUFFER+56                     ; 2193
                    3C   AE        E4   A9  7D 000DD        MOVQ    OUTPUT_ATTBUF+24, BUFFER+60                     ; 2194
                    44   AE        1E   A9  B0 000E2        MOVW    OUTPUT_ATTBUF+82, BUFFER+68                     ; 2196
                    46   AE        18   A9  B0 000E7        MOVW    OUTPUT_ATTBUF+76, BUFFER+70                     ; 2198
                    48   AE        2C   A9  7D 000EC        MOVQ    OUTPUT_ATTBUF+96, BUFFER+72                     ; 2199
                    50   AE        34   A9  7D 000F1        MOVQ    OUTPUT_ATTBUF+104, BUFFER+80                    ; 2201
                         50        B8   A9  D0 000F6        MOVL    OUTPUT_BAD, R0                                  ; 2203
                    C8   AD        04   A0  D0 000FA        MOVL    4(R0), BUFFER+456                               ;
 0C          20          6E             00  2C 000FF        MOVC5   #0, (SP), #32, #12, BUFFER+460                  ; 2204
                                   CC   AD     00104
                                        57  D5 00106        TSTL    R7                                              ; 2205
                                        07  13 00108        BEQL    4$                                              ;
       CC    AD   FED0   C9             0C  28 0010A        MOVC3   #12, COM_O_STRUCNAME, BUFFER+460                ; 2207
 0C          20     D0   B9        CC   A9  2C 00111 4$:    MOVC5   OUTPUT_ATTBUF, @OUTPUT_ATTBUF+4, #32, #12, -    ; 2212
                                   D8   AD     00118                BUFFER+472
 0C          20     D8   B9        D4   A9  2C 0011A        MOVC5   OUTPUT_ATTBUF+8, @OUTPUT_ATTBUF+12, #32, -      ; 2217
                                   E4   AD     00121                #12, BUFFER+484
       F0    AD   FE98   CF             0C  28 00123        MOVC3   #12, P.AAF, BUFFER+496                          ; 2218
                         52             56  D0 0012A        MOVL    R6, J                                           ; 2219
                                        07  11 0012D        BRB     6$                                              ;
                                        5E  DD 0012F 5$:    PUSHL   SP                                              ; 2220
                 0000V   CF             01  FB 00131        CALLS   #1, WRITE_HOMEBLOCK                             ;
                         F6             52  F5 00136 6$:    SOBGTR  J, 5$                                           ;
                         6E        40   AB  D0 00139        MOVL    64(P$), BUFFER                                  ; 2221
             52     14   AB             01  C1 0013D        ADDL3   #1, 20(P$), J                                   ; 2222
                                        07  11 00142        BRB     8$                                              ;
                                        5E  DD 00144 7$:    PUSHL   SP                                              ; 2223
                 0000V   CF             01  FB 00146        CALLS   #1, WRITE_HOMEBLOCK                             ;
                         F6             52  F5 0014B 8$:    SOBGTR  J, 7$                                           ;
                         6E        44   AB  D0 0014E        MOVL    68(P$), BUFFER                                  ; 2224
```

```
              52        14   AB               01   C1 00152         ADDL3   #1, 20(P$), J                              ; 2225
                                              07   11 00157         BRB     10$                                        ;
                                              5E   DD 00159 9$:     PUSHL   SP                                         ; 2226
                     0000V   CF               01   FB 0015B         CALLS   #1, WRITE_HOMEBLOCK                        ;
                             F6               52   F5 00160 10$:    SOBGTR  J, 9$                                      ;
                             52         18    AB   D0 00163         MOVL    24(P$), R2                                 ; 2232
                        07   A2               02   88 00167         BISB2   #2, 7(R2)                                  ;
                             50         0E    AE   3C 0016B         MOVZWL  BUFFER+14, R0                              ; 2233
                        04   A2               50   B0 0016F         MOVW    R0, 4(R2)                                  ;
                             53         20    AE   3C 00173         MOVZWL  BUFFER+32, R3                              ; 2234
                             51             6340   DE 00177         MOVAL   (R3)[R0], R1                               ;
                        1A   A2               51   B0 0017B         MOVW    R1, 26(R2)                                 ;
         1C   A2             53               0C   78 0017F         ASHL    #12, R3, 28(R2)                            ; 2235
                        14   A2         4C    AB   D0 00184         MOVL    76(P$), 20(R2)                             ; 2236
              7E             53               09   78 00189         ASHL    #9, R3, -(SP)                              ; 2242
                 00000000G   00               01   FB 0018D         CALLS   #1, GET_ZERO_VM                            ;
                        10   A2               50   D0 00194         MOVL    R0, 16(R2)                                 ;
              08        07   A2               03   E1 00198         BBC     #3, 7(R2), 11$                             ; 2243
                        10   B2       01FF    8F   3C 0019D         MOVZWL  #511, @16(R2)                              ; 2244
                                              06   11 001A3         BRB     12$                                        ;
                        10   B2       0107    8F   3C 001A5 11$:    MOVZWL  #263, @16(R2)                              ; 2245
              56             53         4C    AB   C1 001AB 12$:    ADDL3   76(P$), R3, R6                             ; 2250
                                              56   D7 001B0         DECL    LBN                                        ;
  0200   8F   00      FD3F   CF       00C8    8F   2C 001B2         MOVC5   #200, INITIAL_HEADER, #0, #512, BUFFER     ; 2251
                                              6E      001BD                                                            ;
                        3C   AE         F4    A9   D0 001BE         MOVL    OUTPUT_ATTBUF+40, BUFFER+60                ; 2255
                        40   AE         12    A9   B0 001C3         MOVW    OUTPUT_ATTBUF+70, BUFFER+64                ; 2256
                        38   AE         14    A9   B0 001C8         MOVW    OUTPUT_ATTBUF+72, BUFFER+56                ; 2257
                        66   AE         E4    A9   7D 001CD         MOVQ    OUTPUT_ATTBUF+24, IDENT_AREA+22            ; 2258
                        6E   AE         E4    A9   7D 001D2         MOVQ    OUTPUT_ATTBUF+24, IDENT_AREA+30            ; 2260
                                              5E   DD 001D7         PUSHL   SP                                         ; 2262
                 00000000G   00               01   FB 001D9         CALLS   #1, CHECKSUM                               ;
                                              5E   DD 001E0         PUSHL   SP                                         ; 2263
                                        09    A6   9F 001E2         PUSHAB  9(LBN)                                     ;
                             6A               02   FB 001E5         CALLS   #2, WRITE_BLOCK                            ;
                        08   AE   00010001    8F   D0 001E8         MOVL    #65537, BUFFER+8                           ; 2268
                        16   AE       0200    8F   B0 001F0         MOVW    #512, BUFFER+22                            ; 2270
                        24   AE       0200    8F   B0 001F6         MOVW    #512, BUFFER+36                            ; 2271
                             50         14    AB   D0 001FC         MOVL    20(P$), R0                                 ; 2273
                             51      30 BB40       DE 00200         MOVAL   @48(P$)[R0], R1                            ;
         18   AE             51               10   9C 00205         ROTL    #16, R1, BUFFER+24                         ;
              51             69   00000FFF    8F   C1 0020A         ADDL3   #4095, OUTPUT_ATTBUF+52, R1                ; 2275
                             51   00001000    8F   C6 00212         DIVL2   #4096, R1                                  ;
                             50             6140   DE 00219         MOVAL   (R1)[R0], R0                               ;
                             50         04    A9   C0 0021D         ADDL2   OUTPUT_ATTBUF+56, R0                       ;
                                              50   D6 00221         INCL    R0                                         ;
         1C   AE             50               10   9C 00223         ROTL    #16, R0, BUFFER+28                         ;
                        4C   AE               50   D0 00228         MOVL    R0, BUFFER+76                              ; 2277
         50   AE      FD9B   CF               06   28 0022C         MOVC3   #6, P.AAG, IDENT_AREA                      ; 2278
                             50               5B   D0 00233         MOVL    P$, R0                                     ; 2279
                             57         20    A0   D0 00236         MOVL    32(R0), MAP_COUNT                          ;
                             53         3C    AB   D0 0023A         MOVL    60(P$), MAP_LBN                            ; 2280
                             52               01   D0 0023E         MOVL    #1, J                                      ; 2281
              50             57               53   C1 00241 13$:    ADDL3   MAP_LBN, MAP_COUNT, R0                     ; 2283
                   3C AB42                    50   D1 00245         CMPL    R0, 60(P$)[J]                              ;
                                              07   12 0024A         BNEQ    14$                                        ;
                             57      20 AB42       C0 0024C         ADDL2   32(P$)[J], MAP_COUNT                       ; 2285
```

```
                                              18  11 00251        BRB     15$                                    :
                                              53  DD 00253 14$:   PUSHL   MAP_LBN                                : 2288
                                              57  DD 00255        PUSHL   MAP_COUNT                              :
                                   08         AE  9F 00257        PUSHAB  BUFFER                                 :
                  00000000G  00               03  FB 0025A        CALLS   #3, MAKE_POINTER                       :
                             57         20 AB42   D0 00261        MOVL    32(P$)[J], MAP_COUNT                   : 2289
                             53         3C AB42   D0 00266        MOVL    60(P$)[J], MAP_LBN                     : 2290
              D2             52               04  F3 0026B 15$:   AOBLEQ  #4, J, 13$                             : 2281
                                              53  DD 0026F        PUSHL   MAP_LBN                                : 2293
                                              57  DD 00271        PUSHL   MAP_COUNT                              :
                                   08         AE  9F 00273        PUSHAB  BUFFER                                 :
                  00000000G  00               03  FB 00276        CALLS   #3, MAKE_POINTER                       :
                                              5E  DD 0027D        PUSHL   SP                                     : 2294
                  00000000G  00               01  FB 0027F        CALLS   #1, CHECKSUM                           :
                                              5E  DD 00286        PUSHL   SP                                     : 2295
                                   01         A6  9F 00288        PUSHAB  1(LBN)                                 :
                             6A               02  FB 0028B        CALLS   #2, WRITE_BLOCK                        :
                             53    14         AB  D0 0028E        MOVL    20(P$), R3                             : 2296
                             52               01  CE 00292        MNEGL   #1, J                                  :
                                              09  11 00295        BRB     17$                                    :
                                              5E  DD 00297 16$:   PUSHL   SP                                     : 2297
                                   48 BB42        9F 00299        PUSHAB  @72(P$)[J]                             :
                             6A               02  FB 0029D        CALLS   #2, WRITE_BLOCK                        :
              F3             52               53  F2 002A0 17$:   AOBLSS  R3, J, 16$                             :
                             7E               01  7D 002A4        MOVQ    #1, -(SP)                              : 2298
                             58    18         AB  D0 002A7        MOVL    24(P$), R8                             :
                                              58  DD 002AB        PUSHL   R8                                     :
                             7E    06         A8  9A 002AD        MOVZBL  6(R8), -(SP)                           :
                                   10         AE  9F 002B1        PUSHAB  BUFFER                                 :
                  00000000G  00               05  FB 002B4        CALLS   #5, CREATE_WINDOW                      :
                             50               68  D0 002BB        MOVL    (R8), R0                               : 2299
                             51    14         AB  D0 002BE        MOVL    20(P$), R1                             :
              52             69 00000FFF      8F  C1 002C2        ADDL3   #4095, OUTPUT_ATTBUF+52, R2            : 2300
                             52 00001000      8F  C6 002CA        DIVL2   #4096, R2                              :
                       0C    A0         0A A241   DE 002D1        MOVAL   10(R2)[R1], 12(R0)                     :
0138    8F    00             6E               00  2C 002D7        MOVC5   #0, (SP), #0, #312, BUFFER+200         : 2305
                                  00C8        CE     002DE                                                       :
                                   3A         AE  94 002E1        CLRB    BUFFER+58                              : 2306
                       08    AE 00030003      8F  D0 002E4        MOVL    #196611, BUFFER+8                      : 2307
          50  AE     FCE1    CF               06  28 002EC        MOVC3   #6, P.AAH, IDENT_AREA                  : 2309
                                              57  D4 002F3        CLRL    MAP_COUNT                              : 2310
              52       B8    A9               08  C1 002F5        ADDL3   #8, OUTPUT_BAD, BAD                    : 2311
                             54    B8         B9  D0 002FA        MOVL    @OUTPUT_BAD, R4                        : 2312
                             53               01  CE 002FE        MNEGL   #1, J                                  :
                                              2B  11 00301        BRB     20$                                    :
                             57    04         A2  C0 00303 18$:   ADDL2   4(BAD), MAP_COUNT                      : 2314
                                              62  DD 00307        PUSHL   (BAD)                                  : 2315
                                   04         A2  DD 00309        PUSHL   4(BAD)                                 :
                                   08         AE  9F 0030C        PUSHAB  BUFFER                                 :
                  00000000G  00               03  FB 0030F        CALLS   #3, MAKE_POINTER                       :
                             12               50  E8 00316        BLBS    R0, 19$                                :
                                   20         A8  9F 00319        PUSHAB  32(R8)                                 : 2316
                                              01  DD 0031C        PUSHL   #1                                     :
                                 00000000G    8F  DD 0031E        PUSHL   #BACKUP$_MAXBAD                        :
                  00000000G  00               03  FB 00324        CALLS   #3, LIB$SIGNAL                         :
                             52               08  C0 0032B 19$:   ADDL2   #8, BAD                                : 2317
              D1             53               54  F2 0032E 20$:   AOBLSS  R4, J, 18$                             : 2312
```

```
            18    AE            57           10  9C 00332         ROTL    #16, MAP_COUNT, BUFFER+24          ; 2319
                                             57  D6 00337         INCL    R7                                 ; 2320
            1C    AE            57           10  9C 00339         ROTL    #16, R7, BUFFER+28
                         4C     AE           57  D0 0033E         MOVL    R7, BUFFER+76                      ; 2321
                                             5E  DD 00342         PUSHL   SP                                 ; 2322
                  00000000G     00           01  FB 00344         CALLS   #1, CHECKSUM
                                             5E  DD 0034B         PUSHL   SP                                 ; 2323
                                       03    A6  9F 0034D         PUSHAB  3(LBN)
                                6A           02  FB 00350         CALLS   #2, WRITE_BLOCK
  0138    8F            00      6E           00  2C 00353         MOVC5   #0, (SP), #0, #312, BUFFER+200     ; 2328
                                     00C8    CE     0035A
                                       3A    AE  94 0035D         CLRB    BUFFER+58                          ; 2329
                         08     AE 00020002  8F  D0 00360         MOVL    #131074, BUFFER+8                  ; 2330
                         34     AE     80    8F  88 00368         BISB2   #128, BUFFER+52                    ; 2332
            18    AE     34     AB           10  9C 0036D         ROTL    #16, 52(P$), BUFFER+24             ; 2333
                  50     0C     AB     14    AB  C7 00373         DIVL3   20(P$), 12(P$), R0                 ; 2334
                                50   0FFF    C0  9E 00379         MOVAB   4095(R0), R0
                                50 00001000  8F  C6 0037E         DIVL2   #4096, R0
                                50           02  C0 00385         ADDL2   #2, R0
            1C    AE            50           10  9C 00388         ROTL    #16, R0, BUFFER+28
                         4C     AE           50  D0 0038D         MOVL    R0, BUFFER+76                      ; 2335
            50    AE   FC42     CF           06  28 00391         MOVC3   #6, P.AAI, IDENT_AREA              ; 2336
                                       50    AB  DD 00398         PUSHL   80(P$)                             ; 2337
                                       34    AB  DD 0039B         PUSHL   52(P$)
                                       08    AE  9F 0039E         PUSHAB  BUFFER
                  00000000G     00           03  FB 003A1         CALLS   #3, MAKE_POINTER
                                             5E  DD 003A8         PUSHL   SP                                 ; 2338
                  00000000G     00           01  FB 003AA         CALLS   #1, CHECKSUM
                                             5E  DD 003B1         PUSHL   SP                                 ; 2339
                                       02    A6  9F 003B3         PUSHAB  2(LBN)
                                6A           02  FB 003B6         CALLS   #2, WRITE_BLOCK
                                50     18    AB  D0 003B9         MOVL    24(P$), R0                         ; 2346
                  01     07     A0           03  E0 003BD         BBS     #3, 7(R0), 21$
                                                 04 003C2         RET
  0138    8F            00      6E           00  2C 003C3  21$:   MOVC5   #0, (SP), #0, #312, BUFFER+200     ; 2353
                                     00C8    CE     003CA
                                       3A    AE  94 003CD         CLRB    BUFFER+58                          ; 2354
                         34     AE     80    8F  8A 003D0         BICB2   #128, BUFFER+52                    ; 2355
                                       18    AE  D4 003D5         CLRL    BUFFER+24                          ; 2356
                         1C     AE 00010000  8F  D0 003D8         MOVL    #65536, BUFFER+28                  ; 2357
                         4C     AE           01  D0 003E0         MOVL    #1, BUFFER+76                      ; 2358
                         08     AE 00070007  8F  D0 003E4         MOVL    #458759, BUFFER+8                  ; 2359
            50    AE   FBED     CF           06  28 003EC         MOVC3   #6, P.AAJ, IDENT_AREA              ; 2361
                                7E   01FE    8F  3C 003F3         MOVZWL  #510, -(SP)                        ; 2362
                                       04    AE  9F 003F8         PUSHAB  BUFFER
                  00000000G     00           02  FB 003FB         CALLS   #2, CHECKSUM2
                                             5E  DD 00402         PUSHL   SP                                 ; 2363
                                       07    A6  9F 00404         PUSHAB  7(LBN)
                                6A           02  FB 00407         CALLS   #2, WRITE_BLOCK
                         08     AE 00050005  8F  D0 0040A         MOVL    #327685, BUFFER+8                  ; 2368
            50    AE   FBCD     CF           06  28 00412         MOVC3   #6, P.AAK, IDENT_AREA              ; 2370
                                7E   01FE    8F  3C 00419         MOVZWL  #510, -(SP)                        ; 2371
                                       04    AE  9F 0041E         PUSHAB  BUFFER
                  00000000G     00           02  FB 00421         CALLS   #2, CHECKSUM2
                                             5E  DD 00428         PUSHL   SP                                 ; 2372
                                       05    A6  9F 0042A         PUSHAB  5(LBN)
                                6A           02  FB 0042D         CALLS   #2, WRITE_BLOCK                    ;
```

```
                  08  AE 00060006  8F  D0 00430        MOVL     #393222, BUFFER+8                        ; 2377
                  16  AE       40  8F  9B 00438        MOVZBW   #64, BUFFER+22                           ; 2379
                  24  AE       40  8F  9B 0043D        MOVZBW   #64, BUFFER+36                           ; 2380
     50  AE     FBA3  CF           06  28 00442        MOVC3    #6, P.AAL, IDENT_AREA                    ; 2381
                  7E       01FE    8F  3C 00449        MOVZWL   #510, -(SP)                              ; 2382
                             04    AE  9F 0044E        PUSHAB   BUFFER                                   ;
          00000000G  00            02  FB 00451        CALLS    #2, CHECKSUM2                            ;
                                   5E  DD 00458        PUSHL    SP                                       ; 2383
                             06    A6  9F 0045A        PUSHAB   6(LBN)                                   ;
                  6A               02  FB 0045D        CALLS    #2, WRITE_BLOCK                          ;
                  08  AE 00080008  8F  D0 00460        MOVL     #524296, BUFFER+8                        ; 2388
                  16  AE     0200  8F  B0 00468        MOVW     #512, BUFFER+22                          ; 2390
                  24  AE     0200  8F  B0 0046E        MOVW     #512, BUFFER+36                          ; 2391
     50  AE     FB77  CF           06  28 00474        MOVC3    #6, P.AAM, IDENT_AREA                    ; 2392
                  7E       01FE    8F  3C 0047B        MOVZWL   #510, -(SP)                              ; 2393
                             04    AE  9F 00480        PUSHAB   BUFFER                                   ;
          00000000G  00            02  FB 00483        CALLS    #2, CHECKSUM2                            ;
                                   5E  DD 0048A        PUSHL    SP                                       ; 2394
                             08    A6  9F 0048C        PUSHAB   8(LBN)                                   ;
                  6A               02  FB 0048F        CALLS    #2, WRITE_BLOCK                          ;
                  08  AE 00040004  8F  D0 00492        MOVL     #262148, BUFFER+8                        ; 2399
                  34  AE     2080  8F  A8 0049A        BISW2    #8320, BUFFER+53                         ; 2402
                  4C  AE           02  D0 004A0        MOVL     #2, BUFFER+76                            ; 2403
                  1C  AE 00020000  8F  D0 004A4        MOVL     #131072, BUFFER+28                       ; 2404
     18  AE       38  AB           10  9C 004AC        ROTL     #16, 56(P$), BUFFER+24                   ; 2405
                  14  AE     0802  8F  B0 004B2        MOVW     #2050, BUFFER+20                         ; 2406
     50  AE     FB39  CF           0A  28 004B8        MOVC3    #10, P.AAN, IDENT_AREA                   ; 2409
                             54    AB  DD 004BF        PUSHL    84(P$)                                   ; 2410
                             38    AB  DD 004C2        PUSHL    56(P$)                                   ;
                             08    AE  9F 004C5        PUSHAB   BUFFER                                   ;
          00000000G  00            03  FB 004C8        CALLS    #3, MAKE_POINTER                         ;
                  7E       01FE    8F  3C 004CF        MOVZWL   #510, -(SP)                              ; 2411
                             04    AE  9F 004D4        PUSHAB   BUFFER                                   ;
          00000000G  00            02  FB 004D7        CALLS    #2, CHECKSUM2                            ;
                                   5E  DD 004DE        PUSHL    SP                                       ; 2412
                             04    A6  9F 004E0        PUSHAB   4(LBN)                                   ;
                  6A               02  FB 004E3        CALLS    #2, WRITE_BLOCK                          ;
                                       04 004E6        RET                                               ; 2415

; Routine Size:  1255 bytes,    Routine Base:  CODE + 0894
```

```
1 %SBTTL 'WRITE_HOMEBLOCK - write home block to volume'
1 ROUTINE WRITE_HOMEBLOCK(BUFFER): L_P$ NOVALUE=
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine computes the checksums in the home block currently
1 !       in the buffer, writes it, and then increments the block numbers
1 !       in the home block for the next write.
1 !
1 ! INPUT PARAMETERS:
1 !       BUFFER          - Pointer to buffer.
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       NONE
1 !
1 !--
1
2 BEGIN
2 MAP
2       BUFFER:         REF BBLOCK;             ! Pointer to buffer
2 L_DECL;
2
2
2 ! Compute the two checksums and then write the block.
2 !
2 CHECKSUM2(.BUFFER, $BYTEOFFSET(HM2$W_CHECKSUM1));
2 CHECKSUM2(.BUFFER, $BYTEOFFSET(HM2$W_CHECKSUM2));
2 WRITE_BLOCK(.BUFFER[HM2$L_HOMELBN], .BUFFER);
2
2
2 ! Advance the block numbers to those of the next home block.
2 !
2 BUFFER[HM2$L_HOMELBN] = .BUFFER[HM2$L_HOMELBN] + 1;
2 BUFFER[HM2$W_HOMEVBN] = .BUFFER[HM2$W_HOMEVBN] + 1;
1 END;


                                000C 00000 WRITE_HOMEBLOCK:
                                                .WORD   Save R2,R3                      ; 2417
                  53 00000000G  00  9E 00002    MOVAB   CHECKSUM2, R3                   ;
                                3A  DD 00009    PUSHL   #58                             ; 2454
```

```
                 52        04   AC  D0 0000B        MOVL    BUFFER, R2                 ;
                                52  DD 0000F        PUSHL   R2                         ;
                 63             02  FB 00011        CALLS   #2, CHECKSUM2              ;
                 7E      01FE   8F  3C 00014        MOVZWL  #510, -(SP)                ; 2455
                                52  DD 00019        PUSHL   R2                         ;
                 63             02  FB 0001B        CALLS   #2, CHECKSUM2              ;
                                52  DD 0001E        PUSHL   R2                         ; 2456
                                62  DD 00020        PUSHL   (R2)                       ;
            F4F3   CF           02  FB 00022        CALLS   #2, WRITE_BLOCK            ;
                                62  D6 00027        INCL    (R2)                       ; 2461
                           10   A2  B6 00029        INCW    16(R2)                     ; 2462
                                    04 0002C        RET                                ; 2463

; Routine Size:  45 bytes,    Routine Base:  CODE + 0D7B
```

```
%SBTTL 'Initial ODS-2 MFD'
!+
!
! The following are the records of the initial MFD for structure level 2.
!
!-

BIND
        INITIAL_MFD = PLIT (
                                                ! MFD (itself)
        WORD (22),                              ! record byte count
        WORD (1),                               ! version limit
        BYTE (0),                               ! flags
        BYTE (10),                              ! name byte count
        BYTE ('000000.DIR'),                    ! name string
        WORD (1),                               ! version number
        WORD (4, 4, 0),                         ! file ID

                                                ! backup journal file
        WORD (22),                              ! record byte count
        WORD (1),                               ! version limit
        BYTE (0),                               ! flags
        BYTE (10),                              ! name byte count
        BYTE ('BACKUP.SYS'),                    ! name string
        WORD (1),                               ! version number
        WORD (8, 8, 0),                         ! file ID

                                                ! bad block file
        WORD (22),                              ! record byte count
        WORD (1),                               ! version limit
        BYTE (0),                               ! flags
        BYTE (10),                              ! name byte count
        BYTE ('BADBLK.SYS'),                    ! name string
        WORD (1),                               ! version number
        WORD (3, 3, 0),                         ! file ID

                                                ! pending bad block log
        WORD (22),                              ! record byte count
        WORD (1),                               ! version limit
        BYTE (0),                               ! flags
        BYTE (10),                              ! name byte count
        BYTE ('BADLOG.SYS'),                    ! name string
        WORD (1),                               ! version number
        WORD (9, 9, 0),                         ! file ID

                                                ! storage bitmap file
        WORD (22),                              ! record byte cout
        WORD (1),                               ! version limit
        BYTE (0),                               ! flags
        BYTE (10),                              ! name byte count
        BYTE ('BITMAP.SYS'),                    ! name string
        WORD (1),                               ! version .umber
        WORD (2, 2, 0),                         ! file ID

                                                ! standard continuation file
        WORD (22),                              ! record byte count
        WORD (1),                               ! v%rsion limit
```

```
        BYTE (0),                       ! flags
        BYTE (10),                      ! name byte count
        BYTE ('CONTIN.SYS'),            ! name string
        WORD (1),                       ! version number
        WORD (7, 7, 0),                 ! file ID

                                        ! core image file
        WORD (22),                      ! record byte count
        WORD (1),                       ! version limit
        BYTE (0),                       ! flags
        BYTE (10),                      ! name byte count
        BYTE ('CORIMG.SYS'),            ! name string
        WORD (1),                       ! version number
        WORD (5, 5, 0),                 ! file ID

                                        ! index file
        WORD (22),                      ! record byte count
        WORD (1),                       ! version limit
        BYTE (0),                       ! flags
        BYTE (10),                      ! name byte count
        BYTE ('INDEXF.SYS'),            ! name string
        WORD (1),                       ! version number
        WORD (1, 1, 0),                 ! file ID

                                        ! volume set list file
        WORD (22),                      ! record byte count
        WORD (1),                       ! version limit
        BYTE (0),                       ! flags
        BYTE (10),                      ! name byte count
        BYTE ('VOLSET.SYS'),            ! name string
        WORD (1),                       ! version number
        WORD (6, 6, 0),                 ! file ID

        WORD (-1)                       ! end marker

        );
```

```
1 %SBTTL 'INIT_MFD - initialize ODS-2 MFD'
1 ROUTINE INIT_MFD : L_P$ NOVALUE =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine writes the initial master file directory.
1 !
1 !
1 ! CALLING SEQUENCE:
1 !       INIT_MFD ()
1 !
1 ! INPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       allocation table
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       initial MFD written
1 !
1 !--
1
2 BEGIN
2
2 LOCAL
2       BUFFER          : BBLOCK [512]; ! I/O buffer
2
2 L_DECL;
2
2
2 ! Simply copy the MFD records into the buffer, zero filled and write it.
2 !
2
2 CH$COPY (.(INITIAL_MFD-4)*4, INITIAL_MFD,
2         0, 512, BUFFER);
2
2 WRITE_BLOCK (.MFD_LBN, BUFFER);
2
1 END;                                  ! end of routine INIT_MFD


                         00000037  00DA8          .LONG   55           ;
                             0016  00DAC P.AAO:   .WORD   22           ;
                             0001  00DAE          .WORD   1            ;
                               00  00DB0          .BYTE   0            ;
```

```
                                     0A  00DB1          .BYTE   10                          ;
 52  49  44  2E  30  30  30  30  30  30  00DB2          .ASCII  \000000.DIR\                ;
                                   0001  00DBC          .WORD   1                           ;
                     0000  0004  0004  00DBE          .WORD   4, 4, 0                     ;
                                   0016  00DC4          .WORD   22                          ;
                                   0001  00DC6          .WORD   1                           ;
                                     00  00DC8          .BYTE   0                           ;
                                     0A  00DC9          .BYTE   10                          ;
 53  59  53  2E  50  55  4B  43  41  42  00DCA          .ASCII  \BACKUP.SYS\                ;
                                   0001  00DD4          .WORD   1                           ;
                     0000  0008  0008  00DD6          .WORD   8, 8, 0                     ;
                                   0016  00DDC          .WORD   22                          ;
                                   0001  00DDE          .WORD   1                           ;
                                     00  00DE0          .BYTE   0                           ;
                                     0A  00DE1          .BYTE   10                          ;
 53  59  53  2E  4B  4C  42  44  41  42  00DE2          .ASCII  \BADBLK.SYS\                ;
                                   0001  00DEC          .WORD   1                           ;
                     0000  0003  0003  00DEE          .WORD   3, 3, 0                     ;
                                   0016  00DF4          .WORD   22                          ;
                                   0001  00DF6          .WORD   1                           ;
                                     00  00DF8          .BYTE   0                           ;
                                     0A  00DF9          .BYTE   10                          ;
 53  59  53  2E  47  4F  4C  44  41  42  00DFA          .ASCII  \BADLOG.SYS\                ;
                                   0001  00E04          .WORD   1                           ;
                     0000  0009  0009  00E06          .WORD   9, 9, 0                     ;
                                   0016  00E0C          .WORD   22                          ;
                                   0001  00E0E          .WORD   1                           ;
                                     00  00E10          .BYTE   0                           ;
                                     0A  00E11          .BYTE   10                          ;
 53  59  53  2E  50  41  4D  54  49  42  00E12          .ASCII  \BITMAP.SYS\                ;
                                   0001  00E1C          .WORD   1                           ;
                     0000  0002  0002  00E1E          .WORD   2, 2, 0                     ;
                                   0016  00E24          .WORD   22                          ;
                                   0001  00E26          .WORD   1                           ;
                                     00  00E28          .BYTE   0                           ;
                                     0A  00E29          .BYTE   10                          ;
 53  59  53  2E  4E  49  54  4E  4F  43  00E2A          .ASCII  \CONTIN.SYS\                ;
                                   0001  00E34          .WORD   1                           ;
                     0000  0007  0007  00E36          .WORD   7, 7, 0                     ;
                                   0016  00E3C          .WORD   22                          ;
                                   0001  00E3E          .WORD   1                           ;
                                     00  00E40          .BYTE   0                           ;
                                     0A  00E41          .BYTE   10                          ;
 53  59  53  2E  47  4D  49  52  4F  43  00E42          .ASCII  \CORIMG.SYS\                ;
                                   0001  00E4C          .WORD   1                           ;
                     0000  0005  0005  00E4E          .WORD   5, 5, 0                     ;
                                   0016  00E54          .WORD   22                          ;
                                   0001  00E56          .WORD   1                           ;
                                     00  00E58          .BYTE   0                           ;
                                     0A  00E59          .BYTE   10                          ;
 53  59  53  2E  46  58  45  44  4E  49  00E5A          .ASCII  \INDEXF.SYS\                ;
                                   0001  00E64          .WORD   1                           ;
                     0000  0001  0001  00E66          .WORD   1, 1, 0                     ;
                                   0016  00E6C          .WORD   22                          ;
                                   0001  00E6E          .WORD   1                           ;
                                     00  00E70          .BYTE   0                           ;
                                     0A  00E71          .BYTE   10                          ;
```

```
          53  59  53  2E  54  45  53  4C  4F  56  00E72        .ASCII  \VOLSET.SYS\                              ;
                                              0001  00E7C        .WORD   1                                         ;
                              0000  0006  0006  00E7E        .WORD   6, 6, 0                                   ;
                                              FFFF  00E84        .WORD   -1                                        ;
                                                    00E86        .BLKB   2

                                                         INITIAL_MFD=           P.AAO

                                        003C 00000 INIT_MFD:
                                                                 .WORD   Save R2,R3,R4,R5                          ; 2558
                          5E       FE00   CE   9E 00002        MOVAB   -512(SP), SP
                 50      FF14     CF       02   78 00007        ASHL    #2, INITIAL_MFD-4, R0                     ; 2601
  0200    8F     00      FF12     CF       50   2C 0000D        MOVC5   R0, INITIAL_MFD, #0, #512, BUFFER         ;
                                           6E      00016                                                          ;
                                           5E   DD 00017        PUSHL   SP                                        ; 2604
                                   54      AB   DD 00019        PUSHL   84(P$)                                    ;
                         F3EC     CF       02   FB 0001C        CALLS   #2, WRITE_BLOCK                           ;
                                                04 00021        RET                                               ; 2606

; Routine Size:  34 bytes,    Routine Base:  CODE + 0E88
```

```
%SBTTL 'INITIALIZE_VOLUME - main volume initialization routine'
GLOBAL ROUTINE INITIALIZE_VOLUME(PVCB,PDEVCHAR): NOVALUE=

!++
!
! FUNCTIONAL DESCRIPTION:
!       This routine initializes an output disk volume during an image restore.
!
! INPUT PARAMETERS:
!       PVCB            - Pointer to VCB for output volume.
!       PDEVCHAR        - Pointer to device characteristics for output volume.
!
! IMPLICIT INPUTS:
!       OUTPUT_ATTBUF   - Contains volume summary attributes.
!       VCB[VCB_CHAN]   - Contains channel number assigned to volume.
!
! OUTPUT PARAMETERS:
!       NONE
!
! IMPLICIT OUTPUTS:
!       NONE
!
! ROUTINE VALUE:
!       NONE
!
! SIDE EFFECTS:
!       NONE
!
!--

BEGIN
LOCAL
        P$AREA:         VECTOR[P$SIZE],         ! Impure area
        ACB:            REF BBLOCK,             ! Pointer to ACB
        C;                                      ! Temp
GLOBAL REGISTER
        P$ = 11:        REF VECTOR;             ! Impure area base register


! Initialize.
!
P$ = P$AREA;
CH$FILL(0, 2*ALLOC_MAX*4, _ALLOC_TABLE_CNT);
_VCB = .PVCB;
_DEVCHAR = .PDEVCHAR;
CLUSTER = .OUTPUT_ATTBUF[VSR_CLUSTER];
STRUCLEV_1 = (.OUTPUT_ATTBUF[VSR_STRUCLEV] EQL 1);


! Strip trailing spaces from the volume name.  This is necessary so that it can
! be zero-filled for an ODS-1 header.
!
DECR J FROM .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$W_LENGTH]-1 TO 0 DO
    BEGIN
    IF .VECTOR[.BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$A_POINTER], .J ;,BYTE] EQL %C' '
    THEN
        BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$W_LENGTH] =
```

```
3                 .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$W_LENGTH] - 1;
2         END;
2
2
2   ! Get the channel number assigned to the device.
2   !
2   CHANNEL = .VCB[VCB_CHAN];
2
2
2   ! Get the bad block information.
2   !
2   OUTPUT_BAD = GET_BADBLOCKS(.VCB[VCB_FAB], .CHANNEL, ._DEVCHAR, .CLUSTER);
2
2
2   ! Compute volume size, rounded up to next cluster boundary.
2   !
2   VOLUME_SIZE = (.DEVCHAR[DIB$L_MAXBLOCK] + .CLUSTER - 1) / .CLUSTER * .CLUSTER;
2
2
2   ! Allocate ACB to describe entire volume.
2   !
2   ACB = GET_VM(ACB_S_ENTRY);
2   VCB[VCB_ACB_FLINK] = .ACB;
2   VCB[VCB_ACB_BLINK] = .ACB;
2   ACB[ACB_FLINK] = VCB[VCB_ACB_FLINK];
2   ACB[ACB_BLINK] = VCB[VCB_ACB_FLINK];
2   ACB[ACB_COUNT] = .VOLUME_SIZE;
2   ACB[ACB_LBN] = 0;
2
2
2   ! Unless the user has preinitialized the volume, increase the maximum number
2   ! of files if warranted by the output volume size.
2   !
2   IF
2       .QUAL[QUAL_INIT] AND
2       .OUTPUT_ATTBUF[VSR_MAXFILES] LSSU
3           .DEVCHAR[DIB$L_MAXBLOCK] / ((.CLUSTER + 1) * 2)
2   THEN
2       OUTPUT_ATTBUF[VSR_MAXFILES] =
2           .DEVCHAR[DIB$L_MAXBLOCK] / ((.CLUSTER + 1) * 2);
2
2
2   ! Now verify the parameters against the volume size and characteristics.
2   !
2   IF .STRUCLEV_1
2   THEN
3       BEGIN
3       IF .DEVCHAR[DIB$L_MAXBLOCK] GTRU 255^12
3       THEN
3           SIGNAL(BACKUP$_LARGECNT, 1, VCB[VCB_DEVICE]);
3       IF .OUTPUT_ATTBUF[VSR_MAXFILES] GTRU 65500
3       THEN OUTPUT_ATTBUF[VSR_MAXFILES] = 65500;
2       END;
2
2
2   ! Check the cluster factor against its lower bound such that the storage map
2   ! does not exceed 255 blocks.  Also check for a reasonable minimum number of
```

```
! clusters.
!
IF
    .VOLUME_SIZE / .CLUSTER GTRU 255*12 OR
    .VOLUME_SIZE / .CLUSTER LSS 50
THEN
    SIGNAL(BACKUP$_CLUSTER, 1, VCB[VCB_DEVICE]);


! Increase maximum number of files if necessary to be input index file size.
!
IF .OUTPUT_ATTBUF[VSR_MAXFILNUM] GTRU .OUTPUT_ATTBUF[VSR_MAXFILES]
THEN
    OUTPUT_ATTBUF[VSR_MAXFILES] = .OUTPUT_ATTBUF[VSR_MAXFILNUM];


! Check maximum number of files against number of clusters.
!
C = .VOLUME_SIZE / (.CLUSTER+1);
IF .OUTPUT_ATTBUF[VSR_MAXFILNUM] GTR .C
THEN
    SIGNAL(BACKUP$_CLUSTER, 1, VCB[VCB_DEVICE]);

IF .OUTPUT_ATTBUF[VSR_MAXFILES] GTR .C
THEN
    OUTPUT_ATTBUF[VSR_MAXFILES] = .C;


! Establish the position of the index file.  If /INITIALIZE is specified, or
! the original volume is not the same size, place it in the middle.  Otherwise,
! propagate the original position.
!
! For middle placement, apply a bias factor equal to the size of a default
! initial MFD so that in the usual case the file structure will be allocated
! as INIT would allocate it.
!
! For propagated placement, subtract out the size of the storage bitmap to
! place the index file bitmap on the same LBN.
!
IF
    .OUTPUT_ATTBUF[VSR_INDEXLBN] EQL 0 OR
    .OUTPUT_ATTBUF[VSR_VOLSIZE] NEQ .DEVCHAR[DIB$L_MAXBLOCK]
THEN
    BEGIN
    OUTPUT_ATTBUF[VSR_INDEXLBN] =
        .DEVCHAR[DIB$L_MAXBLOCK] / 2 + MAXU(2, .CLUSTER);
    END
ELSE
    BEGIN
    LOCAL
        T;

    T = (((.VOLUME_SIZE/.CLUSTER + 4095) / 4096) + .CLUSTER) / .CLUSTER * .CLUSTER;
    IF .T LEQU .OUTPUT_ATTBUF[VSR_INDEXLBN]
        THEN OUTPUT_ATTBUF[VSR_INDEXLBN] = .OUTPUT_ATTBUF[VSR_INDEXLBN] - .T
        ELSE OUTPUT_ATTBUF[VSR_INDEXLBN] = 0;
    END;
```

```
2
2
2 ! Now call the routines that do the work of initializing.
2 !
2 INIT_ALLOCATE();
2 INIT_BITMAP();
2 IF .STRUCLEV_1 THEN INIT_INDEX1() ELSE INIT_INDEX();
2 IF .VCB[VCB_SAVESET] THEN INIT_MFD();
2
2
2 ! Note that the initialization is complete.  Record the volume label.
2 !
2 VCB[VCB_INIT_HDRS] = ..VCB[VCB_IMAP];
2 VCB[VCB_INIT_DONE] = TRUE;
2 CH$COPY(
2     .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$W_LENGTH],
2     .BBLOCK[OUTPUT_ATTBUF[VSR_VOLNAME], DSC$A_POINTER],
2     %C' ',
2     HM2$S_VOLNAME,
2     VCB[VCB_VOLNAME]);
2
2
2 ! Release the bad block descriptor.
2 !
2 FREE_VM(BAD_S_HEADER + .OUTPUT_BAD[BAD_NUMDESC] * BAD_S_DESC, .OUTPUT_BAD);
2 OUTPUT_BAD = 0;
1 END;



                                      0BFC 00000         .ENTRY  INITIALIZE_VOLUME, Save R2,R3,R4,R5,R6,R7,-  ; 2608
                                                                 R8,R9,R11                                     ;
                     59 00000000G  8F   D0 00002         MOVL    #BACKUP$_CLUSTER, R9                          ;
                     58 00000000G  00   9E 00009         MOVAB   LIB$SIGNAL, R8                                ;
                     57 00000000'  EF   9E 00010         MOVAB   OUTPUT_ATTBUF+52, R7                          ;
                     5E        A8  AE   9E 00017         MOVAB   -88(SP), SP                                   ;
                     5B            6E   9E 0001B         MOVAB   P$AREA, P$                                    ; 2648
   38        00      6E            00   2C 0001E         MOVC5   #0, (SP), #0, #56, 32(P$)                     ; 2649
                               20  AB      00023                                                           
              18     AB        04  AC   D0 00025         MOVL    PVCB, 24(P$)                                  ; 2650
                     56        1C  AB   9E 0002A         MOVAB   28(P$), R6                                    ;
                     66        08  AC   D0 0002E         MOVL    PDEVCHAR, (R6)                                ; 2651
                     54        14  AB   9E 00032         MOVAB   20(P$), R4                                    ;
                     64        1A  A7   3C 00036         MOVZWL  OUTPUT_ATTBUF+78, (R4)                        ; 2652
                                   50   D4 0003A         CLRL    R0                                            ; 2653
                     01        0D  A7   91 0003C         CMPB    OUTPUT_ATTBUF+65, #1                          ;
                                   02   12 00040         BNEQ    1$                                            ;
                                   50   D6 00042         INCL    R0                                            ;
              04     AB            50   D0 00044 1$:     MOVL    R0, 4(P$)                                     ;
                     50        CC  A7   3C 00048         MOVZWL  OUTPUT_ATTBUF, J                              ; 2659
                                   0A   11 0004C         BRB     3$                                            ;
                     20     D0 B740   91 0004E 2$:       CMPB    @OUTPUT_ATTBUF+4[J], #32                      ; 2661
                                   03   12 00053         BNEQ    3$                                            ;
                               CC  A7   B7 00055         DECW    OUTPUT_ATTBUF                                 ; 2664
                     F3            50   F4 00058 3$:     SOBGEQ  J, 2$                                         ; 2659
```

```
                         50        18   AB   D0 0005B          MOVL     24(P$), R0                                    ; 2670
                         6B        08   A0   3C 0005F          MOVZWL   8(R0), (P$)                                   :
                                        64   DD 00063          PUSHL    (R4)                                          ; 2675
                         52             66   D0 00065          MOVL     (R6), R2                                      :
                                        52   DD 00068          PUSHL    R2                                            :
                                        6B   DD 0006A          PUSHL    (P$)                                          :
                         50        18   AB   D0 0006C          MOVL     24(P$), R0                                    :
                                   30   A0   DD 00070          PUSHL    48(R0)                                        :
             00000000G   00             04   FB 00073          CALLS    #4, GET_BADBLOCKS                             :
                    B8   A7             50   D0 0007A          MOVL     R0, OUTPUT_BAD                                :
        52          70   A2             64   C1 0007E          ADDL3    (R4), 112(R2), R2                             ; 2680
                         52                  D7 00083          DECL     R2                                            :
                         52             64   C6 00085          DIVL2    (R4), R2                                      :
  0C    AB               52             64   C5 00088          MULL3    (R4), R2, 12(P$)                              :
                                        10   DD 0008D          PUSHL    #16                                           ; 2685
             00000000G   00             01   FB 0008F          CALLS    #1, GET_VM                                    :
                         52        18   AB   D0 00096          MOVL     24(P$), R2                                    ; 2686
                    28   A2             50   D0 0009A          MOVL     ACB, 40(R2)                                   :
                    2C   A2             50   D0 0009E          MOVL     ACB, 44(R2)                                   ; 2687
                         60        28   A2   9E 000A2          MOVAB    40(R2), (ACB)                                 ; 2688
                    04   A0        28   A2   9E 000A6          MOVAB    40(R2), 4(ACB)                                ; 2689
                    08   A0        0C   AB   D0 000AB          MOVL     12(P$), 8(ACB)                                ; 2690
                                   0C   A0   D4 000B0          CLRL     12(ACB)                                       ; 2691
        27        FE4E   C7             05   E1 000B3          BBC      #5, QUAL+10, 4$                               ; 2698
                         51             66   D0 000B9          MOVL     (R6), R1                                      ; 2700
                         50             64   D0 000BC          MOVL     (R4), R0                                      :
                         50             02   C4 000BF          MULL2    #2, R0                                        :
                         50             02   C0 000C2          ADDL2    #2, R0                                        :
        51          70   A1             50   C7 000C5          DIVL3    R0, 112(R1), R1                               :
                         51             67   D1 000CA          CMPL     OUTPUT_ATTBUF+52, R1                          :
                                        11   1E 000CD          BGEQU    4$                                            :
                         51             66   D0 000CF          MOVL     (R6), R1                                      ; 2703
                         50             64   D0 000D2          MOVL     (R4), R0                                      :
                         50             02   C4 000D5          MULL2    #2, R0                                        :
                         50             02   C0 000D8          ADDL2    #2, R0                                        :
        67          70   A1             50   C7 000DB          DIVL3    R0, 112(R1), OUTPUT_ATTBUF+52                 :
                         29        04   AB   E9 000E0    4$:   BLBC     4(P$), 6$                                     ; 2708
                         50             66   D0 000E4          MOVL     (R6), R0                                      ; 2711
             000FF000    8F        70   A0   D1 000E7          CMPL     112(R0), #1044480                             :
                                        0E   1B 000EF          BLEQU    5$                                            :
                                   20   A2   9F 000F1          PUSHAB   32(R2)                                        ; 2713
                                        01   DD 000F4          PUSHL    #1                                            :
                           00000000G    8F   DD 000F6          PUSHL    #BACKUP$_LARGECNT                             :
                         68             03   FB 000FC          CALLS    #3, LIB$SIGNAL                                :
             0000FFDC    8F             67   D1 000FF    5$:   CMPL     OUTPUT_ATTBUF+52, #65500                      ; 2714
                                        05   1B 00106          BLEQU    6$                                            :
                         67      FFDC   8F   3C 00108          MOVZWL   #65500, OUTPUT_ATTBUF+52                      ; 2715
        55          0C   AB             64   C7 0010D    6$:   DIVL3    (R4), 12(P$), R5                              ; 2724
             000FF000    8F             55   D1 00112          CMPL     R5, #1044480                                  :
                                        05   1A 00119          BGTRU    7$                                            :
                         32             55   D1 0011B          CMPL     R5, #50                                       ; 2725
                                        0A   18 0011E          BGEQ     8$                                            :
                                   20   A2   9F 00120    7$:   PUSHAB   32(R2)                                        ; 2727
                                        01   DD 00123          PUSHL    #1                                            :
                                        59   DD 00125          PUSHL    R9                                            :
                         68             03   FB 00127          CALLS    #3, LIB$SIGNAL                                :
                         67        04   A7   D1 0012A    8$:   CMPL     OUTPUT_ATTBUF+56, OUTPUT_ATTBUF+52            ; 2732
```

```
                                    04   1B 0012E        BLEQU   9$                                               ;
                        67    04    A7   D0 00130        MOVL    OUTPUT_ATTBUF+56, OUTPUT_ATTBUF+52               ; 2734
           50           64          01   C1 00134 9$:    ADDL3   #1, (R4), R0                                     ; 2739
           53     0C    AB          50   C7 00138        DIVL3   R0, 12(P$), C                                    ;
                        53    04    A7   D1 0013D        CMPL    OUTPUT_ATTBUF+56, C                              ; 2740
                                    0A   15 00141        BLEQ    10$                                              ;
                              20    A2   9F 00143        PUSHAB  32(R2)                                           ; 2742
                                    01   DD 00146        PUSHL   #1                                               ;
                                    59   DD 00148        PUSHL   R9                                               ;
                        68          03   FB 0014A        CALLS   #3, LIB$SIGNAL                                   ;
                        53          67   D1 0014D 10$:   CMPL    OUTPUT_ATTBUF+52, C                              ; 2744
                                    03   15 00150        BLEQ    11$                                              ;
                        67          53   D0 00152        MOVL    C, OUTPUT_ATTBUF+52                              ; 2746
                        52    20    A7   D0 00155 11$:   MOVL    OUTPUT_ATTBUF+84, R2                             ; 2761
                                    0A   13 00159        BEQL    12$                                              ;
                        50          66   D0 0015B        MOVL    (R6), R0                                         ; 2762
                  70    A0    F8    A7   D1 0015E        CMPL    OUTPUT_ATTBUF+44, 112(R0)                        ;
                                    1A   13 00163        BEQL    14$                                              ;
                        50          66   D0 00165 12$:   MOVL    (R6), R0                                         ; 2766
           50     70    A0          02   C7 00168        DIVL3   #2, 112(R0), R0                                  ;
                        51          64   D0 0016D        MOVL    (R4), R1                                         ;
                        02          51   D1 00170        CMPL    R1, #2                                           ;
                                    03   1E 00173        BGEQU   13$                                              ;
                        51          02   D0 00175        MOVL    #2, R1                                           ;
     20    A7           50          51   C1 00178 13$:   ADDL3   R1, R0, OUTPUT_ATTBUF+84                         ;
                                    23   11 0017D        BRB     16$                                              ; 2760
                        50  0FFF    C5   9E 0017F 14$:   MOVAB   4095(R5), R0                                     ; 2773
                        50 00001000 8F   C6 00184        DIVL2   #4096, R0                                        ;
                        50          64   C0 0018B        ADDL2   (R4), R0                                         ;
                        50          64   C6 0018E        DIVL2   (R4), R0                                         ;
                        50          64   C4 00191        MULL2   (R4), T                                          ;
                        52          50   D1 00194        CMPL    T, R2                                            ; 2774
                                    06   1A 00197        BGTRU   15$                                              ;
                  20    A7          50   C2 00199        SUBL2   T, OUTPUT_ATTBUF+84                              ; 2775
                                    03   11 0019D        BRB     16$                                              ;
                              20    A7   D4 0019F 15$:   CLRL    OUTPUT_ATTBUF+84                                 ; 2776
                F131    CF          00   FB 001A2 16$:   CALLS   #0, INIT_ALLOCATE                                ; 2782
                F285    CF          00   FB 001A7        CALLS   #0, INIT_BITMAP                                  ; 2783
                        07    04    AB   E9 001AC        BLBC    4(P$), 17$                                       ; 2784
                F3AB    CF          00   FB 001B0        CALLS   #0, INIT_INDEX1                                  ;
                                    05   11 001B5        BRB     18$                                              ;
                F82E    CF          00   FB 001B7 17$:   CALLS   #0, INIT_INDEX                                   ;
                        50    18    AB   D0 001BC 18$:   MOVL    24(P$), R0                                       ; 2785
           05     07    A0          03   E1 001C0        BBC     #3, 7(R0), 19$                                   ;
                FE14    CF          00   FB 001C5        CALLS   #0, INIT_MFD                                     ;
                        50    18    AB   D0 001CA 19$:   MOVL    24(P$), R0                                       ; 2790
                  18    A0    10    B0   B0 001CE        MOVW    @16(R0), 24(R0)                                  ;
                  07    A0          04   88 001D3        BISB2   #4, 7(R0)                                        ; 2791
 0C        20     D0    B7    CC    A7   2C 001D7        MOVC5   OUTPUT_ATTBUF, @OUTPUT_ATTBUF+4, #32, #12, -     ; 2797
                              38    A0      001DE                56(R0)                                           ;
                              B8    A7   DD 001E0        PUSHL   OUTPUT_BAD                                       ; 2802
                        50    B8    B7   D0 001E3        MOVL    @OUTPUT_BAD, R0                                  ;
           7E           50          03   78 001E7        ASHL    #3, R0, -(SP)                                    ;
                        6E          08   C0 001EB        ADDL2   #8, (SP)                                         ;
           00000000G    00          02   FB 001EE        CALLS   #2, FREE_VM                                      ;
                              B8    A7   D4 001F5        CLRL    OUTPUT_BAD                                       ; 2803
                                         04 001F8        RET                                                      ; 2804
```

```
; Routine Size:  505 bytes,    Routine Base:  CODE + 0EAA
```

```
; 1717          2805  1 END
; 1718          2806  0 ELUDOM



                                                                            .EXTRN  LIB$SIGNAL

;                                  PSECT SUMMARY
;
;       Name                   Bytes                     Attributes
;
;   COMMON                         2124  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  OVR,NOPIC,ALIGN(2)
;   CODE                           4259  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       129         0      1000          00:02.1



;                                   COMMAND QUALIFIERS

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

; Size:           3576 code + 2807 data bytes
; Run Time:          01:18.0
; Elapsed Time:      04:09.3
; Lines/CPU Min:      2157
; Lexemes/CPU-Min: 32869
; Memory Used:  524 pages
; Compilation Complete
```

0016 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

TAPEUTIL
LIS

STAINIVOL
LIS

UTILITY
LIS

STAIOSYS
LIS

STAPARSE
LIS