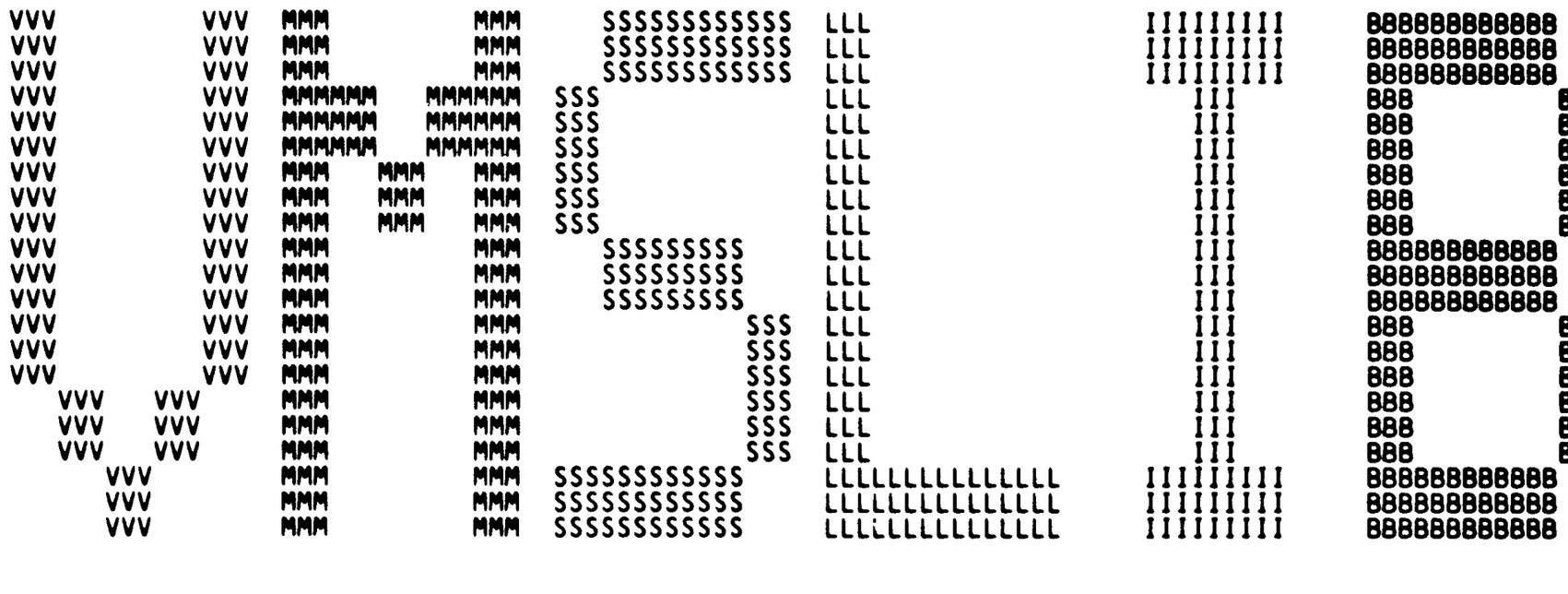
VVV VVV MMM MMM SSSSSSSSSSS LLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSSSS LLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSSSS LLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMMMMM MMMMMM SSS LLL III BBB BBB
VVV VVV MMMMMM MMMMMM SSS LLL III BBB BBB
VVV VVV MMMMMM MMMMMM SSS LLL III BBB BBB
VVV VVV MMM MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSSSSSSSS LLL III BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSS LLL III BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSS LLL III BBBBBBBBBBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSSSSSSSSSS LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSSSS LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBB
VVV MMM MMM SSSSSSSSSSS LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBB

```
FFFFFFFFFF      11          11      DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
FFFFFFFFFF      11          11      DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
FF            1111        1111      DD      DD  EE          FF
FF            1111        1111      DD      DD  EE          FF
FF              11          11      DD      DD  EE          FF
FF              11          11      DD      DD  EE          FF
FFFFFFFF        11          11      DD      DD  EEEEEEEE    FFFFFFFF
FFFFFFFF        11          11      DD      DD  EEEEEEEE    FFFFFFFF
FF              11          11      DD      DD  EE          FF
FF              11          11      DD      DD  EE          FF
FF              11          11      DD      DD  EE          FF              ....
FF              11          11      DD      DD  EE          FF              ....
FF            111111      111111    DDDDDDDD    EEEEEEEEEE  FF              ....
FF            111111      111111    DDDDDDDD    EEEEEEEEEE  FF              ....

  SSSSSSSS  DDDDDDDD    LL
  SSSSSSSS  DDDDDDDD    LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
  SSSSSS    DD      DD  LL
  SSSSSS    DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
```

```
{ F11DEF.SDL - Files-11 on disk structure definitions
{
{ Version:      'V04-000'
{
{*****************************************************************************
{*                                                                           *
{*  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:  VAX/VMS System Macro Libraries
{
{ ABSTRACT:
{
{       This file contains the SDL source for all user visible operating
{       system interfaces.
{
{ ENVIRONMENT:
{
{       n/a
{
{--
{
{
{ AUTHOR:  Andrew C. Goldstein   CREATION DATE:  1-Aug-1976
{
{ MODIFIED BY:
{
{       V03-013 ROW0405         Ralph O. Weber          23-JUL-1984
{               Add two SCB fields for shadowing BACKREV, a BACKUP utility
{               revision number, and GENERNUM, the shadow set generation
{               number.
{
{       V03-012 LMP0197         L. Mark Pilant,         28-Feb-1984  11:53
{               Add a classification enable flag to the volume home block
{               characteristics.
{
```

```
{       V03-011 JWT0139         Jim Teague              9-Nov-1983
{               Change the name of RU qualifiers to be more similar
{               to their SET FILE keyword counterparts.
{
{       V03-010 CDS0002         Christian D. Saether    2-Aug-1983
{               Add SCB$T_VOLOCKNAME, SCB$Q_MOUNTTIME fields.
{
{       V03-009 ACG0339         Andrew C. Goldstein,    3-Jun-1983  15:50
{               Add file structures for non-discretionary protection;
{               Add NORU bit, fix consistency of separate journal bits.
{
{       V03-008 STJ3080         Steven T. Jeffreys,     30-Mar-1983
{               - Decreased fill area at end of $FH2DEF to compensate
{                 for new (STJ3074) field.
{
{       V03-007 STJ3074         Steven T. Jeffreys,     25-Mar-1983
{               - Added FH2$L_HIGHWATER
{               - Added HM2$V_NOHIGHWATER
{
{       V03-006 CDS0001         Christian D. Saether    6-Jan-1983
{               Add SCB$W_WRITECNT.
{
{       V03-005 ACG0302         Andrew C. Goldstein,    3-Dec-1982  14:43
{               Add extended file name area to ident area
{
{       V03-004 JWH0002         Jeffrey W. Horn         16-Sep-1982
{               Fix bug in JWH0001 by switching the AI and BI journaling
{               bits.
{
{       V03-003 JWH0001         Jeffrey W. Horn         12-Jul-1982
{               Modify the definition of the file journaling bits.
{
{       V03-002 LMP0036         L. Mark Pilant          2-Jul-1982  15:35
{               Add an additional file characteristic to indicate that the
{               Access Control List in the header is invalid.
{
{       V03-001 ACG0281         Andrew C. Goldstein,    5-Apr-1982  13:25
{               Add ODS-1 version 2
{
{       V02-015 GAS0038         Gerry Smith             29-Dec-1982
{               Add FAT$W_GBC, the global buffer count for RMS files.
{               This word was added to the RMS attribute area of file headers.
{
{       V02-014 ACG0230         Andrew C. Goldstein,    1-Dec-1981  11:56
{               Add retention date fields to home block
{
{       V02-013 ACG0208         Andrew C. Goldstein,    30-Nov-1981  13:44
{               Add max name length symbol
{
{       V02-013 GAS0019         Gerry Smith             24-Nov-1981
{               Add nobackup bit to Structure Level 1 file header
{
{       V02-012 RAS0028         Ron Schaefer            25-Aug-1981
{               Change definition of FAT$C_STREAM11 to FAT$C_STREAM.
{
{       V02-011 ACG0214         Andrew C. Goldstein,    18-Aug-1981  16:11
```

```
{                Add erase-on-delete bit to file header
{
{       V02-010 RAS0016         Ron Schaefer            10-Aug-1981
{               Add stream record format codes to $FATDEF.
{
{       V02-009 ACG0190         Andrew C. Goldstein,    13-Feb-1981  17:22
{               Add NOCHARGE and journal attributes to file header
{
{       0108    MLJ0001         Martin L. Jack,         10-Nov-1980
{               Add alternate format file ID in directory entry
{
{       0107    ACG0103         Andrew C. Goldstein,    8-Jan-1980  14:46
{               Add backup copies of storage control block flags
{
{       0106    ACG0074         Andrew C. Goldstein,    16-Oct-1979  17:59
{               Changes for back links and additional characteristics
{
{       0105    MCN0003         Maria del C. Nasr       15-Oct-1979   15:03
{               Add HDR3 for magtape file
{
{       0104    ACG0065         Andrew C. Goldstein,    5-Oct-1979  14:13
{               Add overdraft facility to disk quotas
{
{       0103    ACG0041         Andrew C. Goldstein,    22-May-1979  15:19
{               Structures for disk quotas
{
{        0102   ACG0008         Andrew C. Goldstein,  15-Dec-1978  20:32
{       Add placement pointer format bits
{
{        0101   ACG0003         Andrew C. Goldstein,  1-Nov-1978  15:30
{       Add structures for multi-volume files (volume set list)
{**
{
{**
```

```
{+
{
{ File header definitions for Files-11 Structure Level 1
{
{-

module $FH1DEF;

aggregate FH1DEF structure prefix FH1$;
    IDOFFSET byte unsigned;                         /* ident area offset in words
    MPOFFSET byte unsigned;                         /* map area offset in words
    FID_OVERLAY union fill;
        FID word unsigned dimension 2;              /* file ID
        FID_FIELDS structure fill;
            FID_NUM word unsigned;                  /* file number
            FID_SEQ word unsigned;                  /* file sequence number
        end FID_FIELDS;
    end FID_OVERLAY;
    STRUCLEV word unsigned;                         /* file structure level
    constant LEVEL1     equals 257  prefix FH1 tag $C;  /* 401 octal = structure level 1
    FILEOWNER_OVERLAY union fill;
        FILEOWNER word unsigned;                    /* file owner UIC
        FILEOWNER_FIELDS structure fill;
            UICMEMBER byte unsigned;                /* UIC member number
            UICGROUP byte unsigned;                 /* UIC group number
        end FILEOWNER_FIELDS;
    end FILEOWNER_OVERLAY;
    FILEPROT_OVERLAY union fill;
        FILEPROT word unsigned;                     /* file protection
        FILEPROT_BITS structure fill;
            SYSPRO bitfield length 4;               /* system protection
            OWNPRO bitfield length 4;               /* owner protection
            GROUPPRO bitfield length 4;             /* group protection
            WORLDPRO bitfield length 4;             /* world protection
        end FILEPROT_BITS;
    end FILEPROT_OVERLAY;
    FILECHAR_OVERLAY union fill;
        FILECHAR word unsigned;                     /* file characteristics
        FILECHAR_FIELDS structure fill;
            USERCHAR_OVERLAY union fill;
                USERCHAR byte unsigned;             /* user controlled characteristics
                USERCHAR_BITS structure fill;
                    FILL_1 bitfield fill prefix FH1DEF tag $$;/* reserved
                    NOBACKUP bitfield;              /* file is not to be backed up
                    FILL_2 bitfield fill prefix FH1DEF tag $$;/* reserved
                    READCHECK bitfield;             /* verify all read operations
                    WRITCHECK bitfield;             /* verify all write operations
                    CONTIGB bitfield;               /* keep file as contiguous as possible
                    LOCKED bitfield;                /* file is deaccess locked
                    CONTIG bitfield;                /* file is contiguous
                end USERCHAR_BITS;
            end USERCHAR_OVERLAY;
            SYSCHAR_OVERLAY union fill;
                SYSCHAR byte unsigned;              /* system controlled characteristics
                SYSCHAR_BITS structure fill;
```

```
                    FILL_3 bitfield length 4 fill prefix FH1DEF tag $$;/* reserved
                    SPOOL bitfield;                         /* intermediate spool file
                    FILL_4 bitfield fill prefix FH1DEF tag $$;/* reserved
                    BADBLOCK bitfield;                      /* file contains bad blocks
                    MARKDEL bitfield;                       /* file is marked for delete
                end SYSCHAR_BITS;
            end SYSCHAR_OVERLAY;
        end FILECHAR_FIELDS;
    end FILECHAR_OVERLAY;
    RECATTR word unsigned dimension 16;                     /* file record attributes
    constant "LENGTH" equals . prefix FH1$ tag K;           /* length of header area
    constant "LENGTH" equals . prefix FH1$ tag C;           /* length of header area
    FILL_5 word dimension 232 fill prefix FH1DEF tag $$;/* rest of file header
    CHECKSUM word unsigned;                                 /* file header checksum
end FH1DEF;

end_module $FH1DEF;

module $FI1DEF;



aggregate FI1DEF structure prefix FI1$;
    FILENAME word unsigned dimension 3;                     /* file name (RAD-50)
    FILETYPE word unsigned;                                 /* file type (RAD-50)
    VERSION word unsigned;                                  /* version number (binary)
    REVISION word unsigned;                                 /* revision number (binary)
    REVDATE character length 7;                             /* revision date (ASCII DDMMMYY)
    REVTIME character length 6;                             /* revision time (ASCII HHMMSS)
    CREDATE character length 7;                             /* creation date (ASCII DDMMMYY)
    CRETIME character length 6;                             /* creation time (ASCII HHMMSS)
    EXPDATE character length 7;                             /* expiration date (ASCII DDMMMYY)
    FILL_1 byte fill prefix FI1DEF tag $$;                  /* dummy byte to round up
    constant "LENGTH" equals . prefix FI1$ tag K;           /* length of ident area
    constant "LENGTH" equals . prefix FI1$ tag C;           /* length of ident area
    MTHDR1 character length 80;                             /* HDR1 of ANSI magnetic tape file
    MTHDR2 character length 80;                             /* HDR2 of ANSI magnetic tape file
    MTHDR3 character length 80;                             /* HDR3 of ANSI magnetic tape file
end FI1DEF;

end_module $FI1DEF;

module $FM1DEF;



aggregate FM1DEF structure prefix FM1$;
    EX_SEGNUM byte unsigned;                                /* extension segment number of this header
    EX_RVN byte unsigned;                                   /* extension relative volume number
    EX_FILNUM word unsigned;                                /* extension file number
    EX_FILSEQ word unsigned;                                /* extension file sequence number
    COUNTSIZE byte unsigned;                                /* retrieval pointer count field size
    LBNSIZE byte unsigned;                                  /* retrieval pointer LBN field size
    INUSE byte unsigned;                                    /* number of retrieval words in use
    AVAIL byte unsigned;                                    /* number of retrieval words available
    constant POINTERS equals . prefix FM1$ tag K;           /* start of retrieval pointers
```

```
    constant POINTERS equals . prefix FM1$ tag C;       /* start of retrieval pointers
    constant "LENGTH" equals . prefix FM1$ tag K;       /* length of map area
    constant "LENGTH" equals . prefix FM1$ tag C;       /* length of map area

                                                        /* retrieval pointer format
end FM1DEF;

aggregate FM1DEF1 structure prefix FM1$;
    HIGHLBN byte unsigned;                              /* high order LBN
    COUNT byte unsigned;                                /* block count
    LOWLBN word unsigned;                               /* low order LBN
end FM1DEF1;

aggregate FM1DEF2 structure prefix FM1$ origin FILL_1;
    PREVHLBN byte unsigned;
    PREVCOUNT byte unsigned;
    PREVLLBN word unsigned;                             /* previous retrieval pointer
    FILL_1 byte fill prefix FM1DEF tag $$;
end FM1DEF2;

end_module $FM1DEF;
```

```
module $FH2DEF;
/*+
/*
/* File header definitions for Files-11 Structure Level 2
/*
/*-



aggregate FH2DEF structure prefix FH2$;
    IDOFFSET byte unsigned;                         /* ident area offset in words
    MPOFFSET byte unsigned;                         /* map area offset in words
    ACOFFSET byte unsigned;                         /* access control list offset in words
    RSOFFSET byte unsigned;                         /* reserved area offset in words
    SEG_NUM word unsigned;                          /* file segment number
    STRUCLEV_OVERLAY union fill;
        STRUCLEV word unsigned;                     /* file structure level
        STRUCLEV_FIELDS structure fill;
            STRUCVER byte unsigned;                 /* file structure version
            STRUCLEV byte unsigned;                 /* principal file structure level
            constant LEVEL1     equals 257  prefix FH2 tag $C;/* 401 octal = structure level 1
            constant LEVEL2     equals 512  prefix FH2 tag $C;/* 1000 octal = structure level 2
        end STRUCLEV_FIELDS;
    end STRUCLEV_OVERLAY;
    FID_OVERLAY union fill;
        FID word unsigned dimension 3;              /* file ID
        FID_FIELDS structure fill;
            FID_NUM word unsigned;                  /* file number
            FID_SEQ word unsigned;                  /* file sequence number
            FID_RVN_OVERLAY union fill;
                FID_RVN word unsigned;              /* relative volume number
                FID_RVN_FIELDS structure fill;
                    FID_RVN byte unsigned;          /* alternate format RVN
                    FID_NMX byte unsigned;          /* alternate format file number extension
                end FID_RVN_FIELDS;
            end FID_RVN_OVERLAY;
        end FID_FIELDS;
    end FID_OVERLAY;
    EXT_FID_OVERLAY union fill;
        EXT_FID word unsigned dimension 3;          /* extension file ID
        EXT_FID_FIELDS structure fill;
            EX_FIDNUM word unsigned;                /* extension file number
            EX_FIDSEQ word unsigned;                /* extension file sequence number
            EX_FIDRVN_OVERLAY union fill;
                EX_FIDRVN word unsigned;            /* extension relative volume number
                EX_FIDRVN_FIELDS structure fill;
                    EX_FIDRVN byte unsigned;        /* alternate format extension RVN
                    EX_FIDNMX byte unsigned;        /* alternate format extension file number extension
                end EX_FIDRVN_FIELDS;
            end EX_FIDRVN_OVERLAY;
        end EXT_FID_FIELDS;
    end EXT_FID_OVERLAY;
    RECATTR word unsigned dimension 16;             /* file record attributes
    FILECHAR_OVERLAY union fill;
        FILECHAR longword unsigned;                 /* file characteristics
        FILECHAR_BITS structure fill;
```

```
        FILL_1 bitfield fill prefix FH2DEF tag $$;  /* reserved
        NOBACKUP bitfield mask;                     /* file is not to be backed up
        WRITEBACK bitfield mask;                    /* file may be write-back cached
        READCHECK bitfield mask;                    /* verify all read operations
        WRITCHECK bitfield mask;                    /* verify all write operations
        CONTIGB bitfield mask;                      /* keep file as contiguous as possible
        LOCKED bitfield mask;                       /* file is deaccess locked
        CONTIG bitfield mask;                       /* file is contiguous
        FILL_2 bitfield length 3 fill prefix FH2DEF tag $$;/* reserved
        BADACL bitfield mask;                       /* ACL is invalid
        SPOOL bitfield mask;                        /* intermediate spool file
        DIRECTORY bitfield mask;                    /* file is a directory
        BADBLOCK bitfield mask;                     /* file contains bad blocks
        MARKDEL bitfield mask;                      /* file is marked for delete
        NOCHARGE bitfield mask;                     /* file space is not to be charged
        ERASE bitfield mask;                        /* erase file contents before deletion
                                                    /* Note: The high 8 bits of this longword
                                                    /* are reserved for user and CSS use.
    end FILECHAR_BITS;
end FILECHAR_OVERLAY;
RECPROT word unsigned;                              /* record protection
MAP_INUSE byte unsigned;                            /* number of map area words in use
ACC_MODE byte unsigned;                             /* least privileged access mode
FILEOWNER_OVERLAY union fill;
    FILEOWNER longword unsigned;                    /* file owner UIC
    FILEOWNER_FIELDS structure fill;
        UICMEMBER word unsigned;                    /* UIC member number
        UICGROUP word unsigned;                     /* UIC group number
    end FILEOWNER_FIELDS;
end FILEOWNER_OVERLAY;
FILEPROT word unsigned;                             /* file protection
BACKLINK_OVERLAY union fill;
    BACKLINK word unsigned dimension 3;             /* back link pointer
    BACKLINK_FIELDS structure fill;
        BK_FIDNUM word unsigned;                    /* back link file number
        BK_FIDSEQ word unsigned;                    /* back link file sequence number
        BK_FIDRVN_OVERLAY union fill;
            BK_FIDRVN word unsigned;                /* back link relative volume number
            BK_FIDRVN_FIELDS structure fill;
                BK_FIDRVN byte unsigned;            /* alternate format back link RVN
                BK_FIDNMX byte unsigned;            /* alternate format back link file number extension
            end BK_FIDRVN_FIELDS;
        end BK_FIDRVN_OVERLAY;
    end BACKLINK_FIELDS;
end BACKLINK_OVERLAY;
JOURNAL_OVERLAY union fill;
    JOURNAL word unsigned;                          /* journal control flags
    JOURNAL_BITS structure fill;
        ONLY_RU bitfield mask;                      /* file is accessible only in recovery unit
        RUJNL bitfield mask;                        /* enable recovery unit journal
        BIJNL bitfield mask;                        /* enable before image journal
        AIJNL bitfield mask;                        /* enable after image journal
        ATJNL bitfield mask;                        /* enable audit trail journal
        NEVER_RU bitfield mask;                     /* file is never accessible in recovery unit
    end JOURNAL_BITS;
end JOURNAL_OVERLAY;
```

```
    FILL_3 word fill prefix FH2DEF tag $$;              /* reserved
    HIGHWATER longword unsigned;                        /* high-water mark in file
    constant "LENGTH" equals . prefix FH2$ tag K;       /* length of header area
    constant "LENGTH" equals . prefix FH2$ tag C;       /* length of header area
    FILL_6 longword dimension 2 fill;                   /* reserved
    CLASS_PROT structure;                               /* security classification mask
        FILL_5 byte dimension 20 fill;                  /* see structure in $CLSDEF
        end CLASS_PROT;
    constant FULL_LENGTH equals . prefix FH2$ tag K;    /* length of full header
    constant FULL_LENGTH equals . prefix FH2$ tag C;    /* length of full header
    FILL_4 byte dimension 402 fill prefix FH2DEF tag $$;/* rest of file header
    CHECKSUM word unsigned;                             /* file header checksum
end FH2DEF;

end_module $FH2DEF;

module $FI2DEF;



aggregate FI2DEF structure prefix FI2$;
    FILENAME character length 20;                       /* file name, type, and version (ASCII)
    REVISION word unsigned;                             /* revision number (binary)
    CREDATE quadword unsigned;                          /* creation date and time
    REVDATE quadword unsigned;                          /* revision date and time
    EXPDATE quadword unsigned;                          /* expiration date and time
    BAKDATE quadword unsigned;                          /* backup date and time
    FILENAMEXT character length 66;                     /* extension file name area
    constant "LENGTH" equals . prefix FI2$ tag K;       /* length of ident area
    constant "LENGTH" equals . prefix FI2$ tag C;       /* length of ident area
    USERLABEL character length 80;                      /* optional user file label
end FI2DEF;

end_module $FI2DEF;

module $FM2DEF;


                                                        /* retrieval pointer type codes
constant(
      PLACEMENT                                         /* 00 = placement control data
    , FORMAT1                                           /* 01 = format 1
    , FORMAT2                                           /* 10 = format 2
    , FORMAT3                                           /* 11 = format 3
    ) equals 0 increment 1  prefix FM2 tag $C;
                                                        /* format of retrieval pointer

aggregate FM2DEF structure prefix FM2$;
    WORD0_OVERLAY union fill;
        WORD0 word unsigned;                            /* first word, of many uses
        constant LENGTH0 equals . prefix FM2$ tag K;    /* length of format 0 (placement)
        constant LENGTH0 equals . prefix FM2$ tag C;    /* length of format 0 (placement)
        WORD0_BITS0 structure fill;
            FILL_1 bitfield length 14 fill prefix FM2DEF tag $$;/* type specific data
            FORMAT bitfield length 2;                   /* format type code
        end WORD0_BITS0;
```

```
        WORD0_BITS1 structure fill;
            EXACT bitfield;                             /* exact placement specified
            ONCYL bitfield;                             /* on cylinder allocation desired
            FILL_2 bitfield length 10 fill prefix FM2DEF tag $$;
            LBN bitfield;                               /* use LBN of next map pointer
            RVN bitfield;                               /* place on specified RVN
        end WORD0_BITS1;
        WORD0_BITS2 structure fill;
            FILL_3 bitfield length 8 fill prefix FM2DEF tag $$;/* low byte described below
            HIGHLBN bitfield length 6;                  /* high order LBN
        end WORD0_BITS2;
        WORD0_BITS3 structure fill;
            COUNT2 bitfield length 14;                  /* format 2 & 3 count field
        end WORD0_BITS3;
        COUNT1 byte unsigned;                           /* format 1 count field
    end WORD0_OVERLAY;
    LOWLBN word unsigned;                               /* format 1 low order LBN
    constant LENGTH1 equals . prefix FM2$ tag K;        /* length of format 1
    constant LENGTH1 equals . prefix FM2$ tag C;        /* length of format 1
end FM2DEF;

aggregate FM2DEF1 structure prefix FM2$;
    FILL_4 byte dimension 2 fill prefix FM2DEF tag $$;
    LBN2 longword unsigned;                             /* format 2 LBN (longword)
    constant LENGTH2 equals . prefix FM2$ tag K;        /* length of format 2
    constant LENGTH2 equals . prefix FM2$ tag C;        /* length of format 2
end FM2DEF1;

aggregate FM2DEF2 structure prefix FM2$;
    FILL_5 byte dimension 2 fill prefix FM2DEF tag $$;
    LOWCOUNT word unsigned;                             /* format 3 low order count
    LBN3 longword unsigned;                             /* format 3 LBN (longword)
    constant LENGTH3 equals . prefix FM2$ tag K;        /* length of format 3
    constant LENGTH3 equals . prefix FM2$ tag C;        /* length of format 3
end FM2DEF2;

end_module $FM2DEF;

module $FCHDEF;


/*+
/*
/* File characteristics bit definitions. These are identical to, and must
/* track, the bits in FILECHAR above, but are defined relative to the file
/* characteristics longword instead of relative to the file header.
/*
/*-



aggregate FCHDEF union prefix FCH$;
    FILL_1 longword fill prefix FCHDEF tag $$;
        FILL_1_BITS structure fill;
            FILL_2 bitfield fill prefix FCHDEF tag $$;  /* reserved
            NOBACKUP bitfield mask;                     /* file is not to be backed up
```

```
            WRITEBACK bitfield mask;                    /* file may be write-back cached
            READCHECK bitfield mask;                    /* verify all read operations
            WRITCHECK bitfield mask;                    /* verify all write operations
            CONTIGB bitfield mask;                      /* keep file as contiguous as possible
            LOCKED bitfield mask;                       /* file is deaccess locked
            CONTIG bitfield mask;                       /* file is contiguous
            FILL_3 bitfield length 3 fill prefix FCHDEF tag $$;/* reserved
            BADACL bitfield mask;                       /* ACL is invalid
            SPOOL bitfield mask;                        /* intermediate spool file
            DIRECTORY bitfield mask;                    /* file is a directory
            BADBLOCK bitfield mask;                     /* file contains bad blocks
            MARKDEL bitfield mask;                      /* file is marked for delete
            NOCHARGE bitfield mask;                     /* file space is not to be charged
            ERASE bitfield mask;                        /* erase file contents before deletion
                                                        /* Note: The high 8 bits of this longword
                                                        /* are reserved for user and CSS use.
        end FILL_1_BITS;
end FCHDEF;

end_module $FCHDEF;

module $FJNDEF;
/*+
/*
/* File journal control bit definitions. These are identical to, and must
/* track, the bits in JOURNAL above, but are defined relative to the journal
/* control longword instead of relative to the file header.
/*
/*-



aggregate FJNDEF union prefix FJN$;
    FILL_1 word fill prefix FJNDEF tag $$;
        FILL_1_BITS structure fill;
            RUACCESS bitfield mask;                     /* file is accessible only in recovery unit
            RUJNL bitfield mask;                        /* enable recovery unit journal
            BIJNL bitfield mask;                        /* enable before image journal
            AIJNL bitfield mask;                        /* enable after image journal
            ATJNL bitfield mask;                        /* enable audit trail journal
            NORU bitfield mask;                         /* file is not accessible in recovery unit
        end FILL_1_BITS;
end FJNDEF;

end_module $FJNDEF;
```

```
module $FATDEF;
/*+
/*
/* Record attributes area as used by FCS and RMS.
/*
/*-


aggregate FATDEF structure prefix FAT$;
    RTYPE_OVERLAY union fill;
        RTYPE byte unsigned;                            /* record type
        RTYPE_BITS structure fill;
            RTYPE bitfield length 4;                    /* record type subfield
            FILEORG bitfield length 4;                  /* file organization
        end RTYPE_BITS;
        constant(
              UNDEFINED                                 /* undefined record type
            , FIXED                                     /* fixed record type
            , "VARIABLE"                                /* variable length
            , VFC                                       /* variable + fixed control
            , STREAM                                    /* RMS-11 (DEC traditional) stream format
            , STREAMLF                                  /* LF-terminated stream format
            , STREAMCR                                  /* CR-terminated stream format
            ) equals 0 increment 1  prefix FAT tag $C;
        constant(
              SEQUENTIAL                                /* sequential organization
            , RELATIVE                                  /* relative organization
            , INDEXED                                   /* indexed organization
            , DIRECT                                    /* direct organization
            ) equals 0 increment 1  prefix FAT tag $C;
    end RTYPE_OVERLAY;
    RATTRIB_OVERLAY union fill;
        RATTRIB byte unsigned;                          /* record attributes
        RATTRIB_BITS structure fill;
            FORTRANCC bitfield mask;                    /* fortran carriage control
            IMPLIEDCC bitfield mask;                    /* implied carriage control
            PRINTCC bitfield mask;                      /* print file carriage control
            NOSPAN bitfield mask;                       /* no spanned records
        end RATTRIB_BITS;
    end RATTRIB_OVERLAY;
    RSIZE word unsigned;                                /* record size in bytes
    HIBLK_OVERLAY union fill;
        HIBLK longword unsigned;                        /* highest allocated VBN
        HIBLK_FIELDS structure fill;
            HIBLKH word unsigned;                       /* high order word
            HIBLKL word unsigned;                       /* low order word
        end HIBLK_FIELDS;
    end HIBLK_OVERLAY;
    EFBLK_OVERLAY union fill;
        EFBLK longword unsigned;                        /* end of file VBN
        EFBLK_FIELDS structure fill;
            EFBLKH word unsigned;                       /* high order word
            EFBLKL word unsigned;                       /* low order word
        end EFBLK_FIELDS;
    end EFBLK_OVERLAY;
```

```
    FFBYTE word unsigned;                                   /* first free byte in EFBLK
    BKTSIZE byte unsigned;                                  /* bucket size in blocks
    VFCSIZE byte unsigned;                                  /* size in bytes of fixed length control for VFC records
    MAXREC word unsigned;                                   /* maximum record size in bytes
    DEFEXT word unsigned;                                   /* default extend quantity
    GBC word unsigned;                                      /* global buffer count
    FILL_1 word dimension 4 fill prefix FATDEF tag $$;      /* spare
    VERSIONS word unsigned;                                 /* default version limit for directory file
    constant "LENGTH" equals . prefix FAT$ tag K;
    constant "LENGTH" equals . prefix FAT$ tag C;

end FATDEF;

end_module $FATDEF;
```

```
module $HM1DEF;
/**
/*
/* Home block definitions for Files-11 Structure Level 1
/*
/*-



aggregate HM1DEF structure prefix HM1$;
    IBMAPSIZE word unsigned;                            /* index file bitmap size, blocks
    IBMAPLBN longword unsigned;                         /* index file bitmap starting LBN
    MAXFILES word unsigned;                             /* maximum ! files on volume
    CLUSTER word unsigned;                              /* storage bitmap cluster factor
    DEVTYPE word unsigned;                              /* disk device type
    STRUCLEV word unsigned;                             /* volume structure level
    constant LEVEL1     equals 257  prefix HM1 tag $C;  /* 401 octal = structure level 1
    constant LEVEL2     equals 258  prefix HM1 tag $C;  /* 402 octal = structure level 1, version 2
    VOLNAME character length 12;                        /* volume name (ASCII)
    FILL_1 byte dimension 4 fill prefix HM1DEF tag $$;  /* spare
    VOLOWNER word unsigned;                             /* volume owner UIC
    PROTECT_OVERLAY union fill;
        PROTECT word unsigned;                          /* volume protection
        PROTECT_BITS structure fill;
            SYSPRO bitfield length 4;                   /* system protection
            OWNPRO bitfield length 4;                   /* owner protection
            GROUPPRO bitfield length 4;                 /* group protection
            WORLDPRO bitfield length 4;                 /* world protection
        end PROTECT_BITS;
    end PROTECT_OVERLAY;
    VOLCHAR word unsigned;                              /* volume characteristics
    FILEPROT word unsigned;                             /* default file protection
    FILL_2 byte dimension 6 fill prefix HM1DEF tag $$;  /* spare
    WINDOW byte unsigned;                               /* default window size
    EXTEND byte unsigned;                               /* default file extend
    LRU_LIM byte unsigned;                              /* default LRU limit
    FILL_3 byte dimension 11 fill prefix HM1DEF tag $$; /* spare
    CHECKSUM1 word unsigned;                            /* first checksum
    CREDATE character length 14;                        /* volume creation date
    FILL_4 byte dimension 382 fill prefix HM1DEF tag $$;/* spare
    SERIALNUM longword unsigned;                        /* pack serial number
    FILL_5 byte dimension 12 fill prefix HM1DEF tag $$; /* reserved
    VOLNAME2 character length 12;                       /* 2nd copy of volume name
    OWNERNAME character length 12;                      /* volume owner name
    FORMAT character length 12;                         /* volume format type
    FILL_6 byte dimension 2 fill prefix HM1DEF tag $$;  /* spare
    CHECKSUM2 word unsigned;                            /* second checksum
end HM1DEF;

end_module $HM1DEF;
```

```
module $HM2DEF;
/*+
/*
/* Home block definitions for Files-11 Structure Level 2
/*
/*-



aggregate HM2DEF structure prefix HM2$;
    HOMELBN longword unsigned;                          /* LBN of home (i.e., this) block
    ALHOMELBN longword unsigned;                        /* LBN of alternate home block
    ALTIDXLBN longword unsigned;                        /* LBN of alternate index file header
    STRUCLEV_OVERLAY union fill;
        STRUCLEV word unsigned;                         /* volume structure level
        STRUCLEV_FIELDS structure fill;
            STRUCVER byte unsigned;                     /* structure version number
            STRUCLEV byte unsigned;                     /* main structure level
            constant LEVEL1     equals 257  prefix HM2 tag $C;/* 401 octal = structure level 1
            constant LEVEL2     equals 512  prefix HM2 tag $C;/* 1000 octal = structure level 2
        end STRUCLEV_FIELDS;
    end STRUCLEV_OVERLAY;
    CLUSTER word unsigned;                              /* storage bitmap cluster factor
    HOMEVBN word unsigned;                              /* VBN of home (i.e., this) block
    ALHOMEVBN word unsigned;                            /* VBN of alternate home block
    ALTIDXVBN word unsigned;                            /* VBN of alternate index file header
    IBMAPVBN word unsigned;                             /* VBN of index file bitmap
    IBMAPLBN longword unsigned;                         /* LBN of index file bitmap
    MAXFILES longword unsigned;                         /* maximum ! files on volume
    IBMAPSIZE word unsigned;                            /* index file bitmap size, blocks
    RESFILES word unsigned;                             /* ! reserved files on volume
    DEVTYPE word unsigned;                              /* disk device type
    RVN word unsigned;                                  /* relative volume number of this volume
    SETCOUNT word unsigned;                             /* count of volumes in set
    VOLCHAR_OVERLAY union fill;
        VOLCHAR word unsigned;                          /* volume characteristics
        VOLCHAR_BITS structure fill;
            READCHECK bitfield mask;                    /* verify all read operations
            WRITCHECK bitfield mask;                    /* verify all write operations
            ERASE bitfield mask;                        /* erase all files on delete
            NOHIGHWATER bitfield mask;                  /* turn off high-water marking
            CLASS_PROT bitfield mask;                   /* enable classification checks on the volume
        end VOLCHAR_BITS;
    end VOLCHAR_OVERLAY;
    VOLOWNER longword unsigned;                         /* volume owner UIC
    SEC_MASK longword unsigned;                         /* volume security mask
    PROTECT word unsigned;                              /* volume protection
    FILEPROT word unsigned;                             /* default file protection
    RECPROT word unsigned;                              /* default file record protection
    CHECKSUM1 word unsigned;                            /* first checksum
    CREDATE quadword unsigned;                          /* volume creation date
    WINDOW byte unsigned;                               /* default window size
    LRU_LIM byte unsigned;                              /* default LRU limit
    EXTEND word unsigned;                               /* default file extend
    RETAINMIN quadword unsigned;                        /* minimum file retention period
    RETAINMAX quadword unsigned;                        /* maximum file retention period
```

```
    REVDATE quadword unsigned;                          /* volume revision date
    MIN_CLASS structure;                                /* volume minimum security class
        FILL_2 byte dimension 20 fill;
        end MIN_CLASS;
    MAX_CLASS structure;                                /* volume maximum security class
        FILL_3 byte dimension 20 fill;
        end MAX_CLASS;
    FILL_1 byte dimension 320 fill prefix HM2DEF tag $$;/* spare
    SERIALNUM longword unsigned;                        /* pack serial number
    STRUCNAME character length 12;                      /* structure (volume set name)
    VOLNAME character length 12;                        /* volume name
    OWNERNAME character length 12;                      /* volume owner name
    FORMAT character length 12;                         /* volume format type
    FILL_2 byte dimension 2 fill prefix HM2DEF tag $$;  /* spare
    CHECKSUM2 word unsigned;                            /* second checksum
end HM2DEF;

end_module $HM2DEF;
```

```
module $DIRDEF;
/**
/*
/* Directory entry structure for Files-11 Structure Level 2
/*
/*-



aggregate DIRDEF structure prefix DIR$;
    SIZE word unsigned;                                 /* size of directory record in bytes
    VERLIMIT word unsigned;                             /* maximum number of versions
    FLAGS_OVERLAY union fill;
        FLAGS byte unsigned;                            /* status flags
        FLAGS_BITS structure fill;
            TYPE bitfield length 3;                     /* directory entry type
            FILL_1 bitfield length 3 fill prefix DIRDEF tag $$;/* reserved
            NEXTREC bitfield;                           /* another record of same name & type follows
            PREVREC bitfield;                           /* another record of same name & type precedes
        end FLAGS_BITS;
                                                        /* directory entry type codes
        constant(
              FID                                       /* normal file ID
            , LINKNAME                                  /* symbolic name
            ) equals 0 increment 1  prefix DIR tag $C;
    end FLAGS_OVERLAY;
    NAMECOUNT_OVERLAY union fill;
        NAMECOUNT byte unsigned;                        /* byte count of name string
        constant "LENGTH" equals . prefix DIR$ tag K;   /* length of directory entry overhead
        constant "LENGTH" equals . prefix DIR$ tag C;   /* length of directory entry overhead
        NAMECOUNT_FIELDS structure fill;
            FILL_2 byte fill prefix DIRDEF tag $$;
            NAME character length 0 tag T;              /* name string
                                                        /* the version numbers and file ID's follow the
                                                        /* variable length name area in the form of a
                                                        /* blockvector. Each entry is as follows:
            constant NAME       equals 80  prefix DIR tag $S;/* maximum length of name string
        end NAMECOUNT_FIELDS;
    end NAMECOUNT_OVERLAY;
end DIRDEF;

aggregate DIRDEF1 structure prefix DIR$;
    VERSION word;                                       /* version number
    FID_OVERLAY union fill;
        FID word unsigned dimension 3;                  /* file ID
        constant VERSION equals . prefix DIR$ tag K;    /* size of each version entry
        constant VERSION equals . prefix DIR$ tag C;    /* size of each version entry
        FID_FIELDS structure fill;
            FID_NUM word unsigned;                      /* file number
            FID_SEQ word unsigned;                      /* file sequence number
            FID_RVN_OVERLAY union fill;
                FID_RVN word unsigned;                  /* relative volume number
                FID_RVN_FIELDS structure fill;
                    FID_RVN byte unsigned;              /* alternate format RVN
                    FID_NMX byte unsigned;              /* alternate format file number extension
```

```
                end FID_RVN_FIELDS;
            end FID_RVN_OVERLAY;
        end FID_FIELDS;
    end FID_OVERLAY;
end DIRDEF1;

aggregate DIRDEF2 union prefix DIR$;
    LINKNAME character;                     /* symbolic link name (counted string)
end DIRDEF2;

end_module $DIRDEF;
```

```
module $SCBDEF;
/*+
/*
/* Format of storage control block, Files-11 Structure Level 2
/*
/*-


aggregate SCBDEF structure prefix SCB$;
    STRUCLEV_OVERLAY union fill;
        STRUCLEV word unsigned;                         /* file structure level
        STRUCLEV_FIELDS structure fill;
            STRUCVER byte unsigned;                     /* file structure version
            STRUCLEV byte unsigned;                     /* principal file structure level
            constant LEVEL2     equals 512  prefix SCB tag $C;/* 1000 octal = structure level 2
        end STRUCLEV_FIELDS;
    end STRUCLEV_OVERLAY;
    CLUSTER word unsigned;                              /* storage map cluster factor
    VOLSIZE longword unsigned;                          /* volume size in logical blocks
    BLKSIZE longword unsigned;                          /* number of physical blocks per logical block
    SECTORS longword unsigned;                          /* number of sectors per track
    TRACKS longword unsigned;                           /* number of tracks per cylinder
    CYLINDER longword unsigned;                         /* number of cylinders
    STATUS_OVERLAY union fill;
        STATUS longword unsigned;                       /* volume status flags
        STATUS_BITS structure fill;
            MAPDIRTY bitfield mask;                     /* storage map is dirty (partially updated)
            MAPALLOC bitfield mask;                     /* storage map is preallocated (lost blocks)
            FILALLOC bitfield mask;                     /* file numbers are preallocated (lost header slots)
            QUODIRTY bitfield mask;                     /* quota file is dirty (partially updated)
            HDRWRITE bitfield mask;                     /* file headers are write back cached
        end STATUS_BITS;
    end STATUS_OVERLAY;
    STATUS2_OVERLAY union fill;
        STATUS2 longword unsigned;                      /* backup status - bits must match those above
        STATUS2_BITS structure fill;
            MAPDIRTY2 bitfield mask;                    /* storage map is dirty (partially updated)
            MAPALLOC2 bitfield mask;                    /* storage map is preallocated (lost blocks)
            FILALLOC2 bitfield mask;                    /* file numbers are preallocated (lost header slots)
            QUODIRTY2 bitfield mask;                    /* quota file is dirty (partially updated)
            HDRWRITE2 bitfield mask;                    /* file headers are write back cached
        end STATUS2_BITS;
    end STATUS2_OVERLAY;
    WRITECNT word unsigned;                             /* count of write access mounters.
    VOLOCKNAME character length 12;                     /* name used for file system serialization on volume.
    MOUNTTIME quadword unsigned;                        /* time of last initial mount.
    BACKREV word unsigned;                              /* BACKUP revision number.
    GENERNUM quadword unsigned;                         /* shadow set revision number.
    reserved byte dimension 510-. fill;                 /* reserved
    CHECKSUM word unsigned;                             /* block checksum
end SCBDEF;

end_module $SCBDEF;
```

```
module $BBMDEF;
/**
/*
/* Bad block map (generated by bad block scan program)
/*
/*-



aggregate BBMDEF structure prefix BBM$;
    COUNTSIZE byte unsigned;                        /* retrieval pointer count field size
    LBNSIZE byte unsigned;                          /* retrieval pointer LBN field size
    INUSE byte unsigned;                            /* number of retrieval words in use
    AVAIL byte unsigned;                            /* number of retrieval words available
    constant POINTERS equals . prefix BBM$ tag K;   /* start of retrieval pointers
    constant POINTERS equals . prefix BBM$ tag C;   /* start of retrieval pointers
    FILL_1 byte dimension 506 fill prefix BBMDEF tag $$;/* pointer space
    CHECKSUM word unsigned;                         /* block checksum

                                                    /* retrieval pointer format
end BBMDEF;

aggregate BBMDEF1 structure prefix BBM$;
    HIGHLBN byte unsigned;                          /* high order LBN
    COUNT byte unsigned;                            /* block count
    LOWLBN word unsigned;                           /* low order LBN
end BBMDEF1;

aggregate BBMDEF2 structure prefix BBM$ origin FILL_2;
    PREVHLBN byte unsigned;
    PREVCOUNT byte unsigned;
    PREVLLBN word unsigned;                                 /* previous retrieval pointer
    FILL_2 byte fill prefix BBMDEF tag $$;
end BBMDEF2;

end_module $BBMDEF;
```

```
module $BBDDEF;
/**
/*
/* Bad block descriptor (generated by formatters for RK06, RM03, et al)
/*
/*-



aggregate BBDDEF structure prefix BBD$;
    SERIAL longword unsigned;                           /* pack serial number
    RESERVED word unsigned;                             /* reserved area (MBZ)
    FLAGS word unsigned;                                /* pack status flags (zero for normal use)
    constant DESCRIPT equals . prefix BBD$ tag K;       /* start of bad block descriptors
    constant DESCRIPT equals . prefix BBD$ tag C;       /* start of bad block descriptors
    FILL_1 byte dimension 500 fill prefix BBDDEF tag $$;
    LASTWORD longword unsigned;                         /* last longword of block

end BBDDEF;

aggregate BBDDEF1 union prefix BBD$;
    BADBLOCK longword unsigned;                         /* individual bad block entry
    constant "ENTRY" equals . prefix BBD$ tag K;
    constant "ENTRY" equals . prefix BBD$ tag C;
        BADBLOCK_BITS structure fill;
            CYLINDER bitfield length 15;                /* cylinder number of bad block
            FILL_2 bitfield fill prefix BBDDEF tag $$;
            SECTOR bitfield length 8;                   /* sector number of bad block
            TRACK bitfield length 7;                    /* track number of bad block
        end BADBLOCK_BITS;
end BBDDEF1;

end_module $BBDDEF;
```

```
module $VSLDEF;
/**
/*
/* Structure of a volume set list file entry. Record 1 contains the volume
/* set name. Record n+1 contains the volume label of RVN n in the volume set.
/*
/*-


aggregate VSLDEF structure prefix VSL$;
    NAME character length 12;                          /* volume name
    FILL_1 byte dimension 52 fill prefix VSLDEF tag $$; /* unused
    constant "LENGTH" equals . prefix VSL$ tag K;
    constant "LENGTH" equals . prefix VSL$ tag C;

end VSLDEF;

end_module $VSLDEF;
```

```
module $PBBDEF;
/**
/*
/* Pending bad block file record format. Each record describes a disk block
/* on which an error has occurred which has not been turned over to the bad
/* block file.
/*
/*-



aggregate PBBDEF structure prefix PBB$;
    FID word unsigned dimension 3;                  /* File ID of containing file
    FLAGS_OVERLAY union fill;
        FLAGS byte unsigned;                        /* status flags
        FLAGS_BITS structure fill;
            READERR bitfield mask;                  /* read error occurred
            WRITERR bitfield mask;                  /* write error occurred
        end FLAGS_BITS;
    end FLAGS_OVERLAY;
    COUNT byte unsigned;                            /* error count
    VBN longword unsigned;                          /* virtual block in file
    LBN longword unsigned;                          /* logical block number
    constant "LENGTH" equals . prefix PBB$ tag K;   /* length of entry
    constant "LENGTH" equals . prefix PBB$ tag C;   /* length of entry

end PBBDEF;

end_module $PBBDEF;
```

```
module $DQFDEF;
/*+
/*
/* Structure of disk quota file record. Each record contains the authorization
/* and usage of a particular UIC for this volume set.
/*
/*-



aggregate DQFDEF structure prefix DQF$;
    FLAGS_OVERLAY union fill;
        FLAGS longword unsigned;                        /* flags longword, containing...
        FLAGS_BITS structure fill;
            ACTIVE bitfield mask;                       /* record contains an active entry
        end FLAGS_BITS;
    end FLAGS_OVERLAY;
    UIC longword unsigned;                              /* UIC of this record
    USAGE longword unsigned;                            /* number of blocks in use
    PERMQUOTA longword unsigned;                        /* permanent disk quota
    OVERDRAFT longword unsigned;                        /* overdraft limit
    FILL_1 longword dimension 3 fill prefix DQFDEF tag $$;/* reserved
    constant "LENGTH" equals . prefix DQF$ tag K;
    constant "LENGTH" equals . prefix DQF$ tag C;
end DQFDEF;

end_module $DQFDEF;
```

SCRSHR
MAP

DSTDEF
SDL

FIIA
SDL

VERIFYMSG
LIS

LIBCQDEF
SDL

FIIDEF
SDL

ACCDEF
MDL

VMSLIB

OBJFMT
SDL