```
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
```

```
_$
Sy
---
NT$
NT$
NT$
NT$
NT$
NT$

NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$
NT$

NT$

NT$
NT$
NT$
NT$
NT$
NT$
NT

NT
NT
NT
NT
NT
NT
PI
```

```
**FILE**ID**RMSUSR

RRRRRRRR    MM      MM    SSSSSSSS  UU      UU    SSSSSSSS  RRRRRRRR
RRRRRRRR    MM      MM    SSSSSSSS  UU      UU    SSSSSSSS  RRRRRRRR
RR      RR  MMMM  MMMM  SS          UU      UU  SS          RR      RR
RR      RR  MMMM  MMMM  SS          UU      UU  SS          RR      RR
RR      RR  MM  MM  MM  SS          UU      UU  SS          RR      RR
RR      RR  MM  MM  MM  SS          UU      UU  SS          RR      RR
RRRRRRRR    MM      MM    SSSSSS    UU      UU    SSSSSS    RRRRRRRR
RRRRRRRR    MM      MM    SSSSSS    UU      UU    SSSSSS    RRRRRRRR
RR  RR      MM      MM          SS  UU      UU          SS  RR  RR
RR  RR      MM      MM          SS  UU      UU          SS  RR  RR
RR    RR    MM      MM          SS  UU      UU          SS  RR    RR        ....
RR    RR    MM      MM          SS  UU      UU          SS  RR    RR        ....
RR      RR  MM      MM  SSSSSSSS    UUUUUUUUUU  SSSSSSSS    RR      RR      ....
RR      RR  MM      MM  SSSSSSSS    UUUUUUUUUU  SSSSSSSS    RR      RR      ....


  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
```

```
RMS

/*


/*-


/**
/*
/*
```

```
{       $begin  rmsusr,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.                  *
{*                                                                           *
{*                                                                           *
{*****************************************************************************
{
```

```
RMS

/*-

/*
/*
/*
```

```
{
{       rms user structure definitions
{
{
{       Modified By:
{
{       V03-029 RAS0325         Ron Schaefer            11-Jul-1984
{               Make NAM$C_MAXRSS[LCL] equal 255; as big as it can get.
{
{       V03-028 JEJ0030         J E Johnson             19-Apr-1984
{               Move SRCHXABS bit so that any existing refs to the previously
{               deleted NOP bits will not collide with it.
{
{       V03-027 JEJ0016         J E Johnson             27-Mar-1984
{               Delete unused NAM$B_NOP bits and add NAM$V_SRCHXABS bit for
{               network search operations. Also add constant to NAM$B_RFS field.
{
{       V03-026 DGB0027         Donald G. Blair         15-Mar-1984
{               Add new fields for ACL's: XAB$L_ACLBUF, XAB$W_ACLSIZ,
{               XAB$W_ACLLEN, XAB$L_ACLCTX, XAB$L_ACL_STATUS, XAB$B_PROT_OPT,
{               and XAB$V_PROPAGATE.
{
{       V03-025 DGB0003         Donald G. Blair         16-Feb-1984
{               Add new fields to $XABPRO to support access mode
{               protected files: XAB$B_PROT_MODE and FAB$V_FILE_MODE.
{               Add NAM$V_SLPARSE.
{
{       V03-024 DAS0004         David Solomon           13-Feb-1984
{               Remove EDL FOP option (erase on delete). Clean up
{               ordering of FILL_x fields.
{
{       V03-023 KBT0576         Keith B. Thompson       5-Aug-1983
{               Add new flags for FSCN and one new item code
{
{       V03-022 RAS0174         Ron Schaefer            28-Jul-1983
{               Security correction:  Delete FAB$B_DSBMSK and FAB$V_UFM;
{               add FAB$B_ACMODES, FAB$V_LNM_MODE and FAB$V_CHAN_MODE.
{
{       V03-021 DAS0003         David Solomon           27-Jul-1983
{               Max journal name length changed from 12 to 16. Change XABJNL
{               JOP bit RUA to ONLY_RU.
{
{       V03-020 RAS0145         Ron Schaefer            14-Apr-1983
{               Add descriptor code for SYS$FILESCAN ($FSCNDEF).
{               Add NOCONCEAL NOP option; add SEARCH_LIST FNB option to NAM;
{               add NEVER_RU XABJNL option; add EDL FOP option.
{               Fold XABACE into an extended XABPRO and add additional
{               XABPRO fields.
{
{       V03-019 RAS0144         Ron Schaefer            12-Apr-1983
{               Add code (RME$C_PPFECHO) to support echo of SYS$INPUT
{               to SYS$OUTPUT.
{
{       V03-018 LJA0067         Laurie J. Anderson      01-Mar-1983
{               Add new field into XABCXR - CXRBFZ - the length of CXRBUF
{               Marked CXFRTE as no longer needed.
```

RMS

end

end

```
{
{       V03-017 JWH0190         Jeffrey W. Horn         21-Feb-1983
{               Add $XABACE, the Access Control Entry XAB.
{
{       V03-016 DAS0002         David Solomon           15-Feb-1983
{               Forgot to up edit level in V03-015.
{
{       V03-015 DAS0002         David Solomon           15-Feb-1983
{               Add new ROP bit ETO - extended terminal operation.
{
{       V03-014 KBT0494         Keith B. Thompson       11-Feb-1983
{               Add SYNCHK bit to name block NOP field
{
{       V03-013 DAS0002         David Solomon           09-Feb-1983
{               Redo XABTRM structure for new terminal I/O support.
{
{       V03-012 TMK0003         Todd M. Katz            03-Feb-1983
{               Eliminate TMK0002.
{
{       V03-011 KPL0003         Peter Lieberwirth       30-Dec-1982
{               Add FAB bits to indicate recovery is taking place.  These
{               bits are only to be set by the recovery routine that is
{               image merged into the RCP.  No FAB macros support this
{               field; the bits must be set at run time by hand.
{
{       V03-010 TMK0002         Todd M. Katz            22-Dec-1982
{               Re-define the ROP bit RAB$V_FDL to also be RAB$V_FGET.
{
{       V03-009 LJA0045         Laurie J. Anderson      17-Dec-1982
{               Add Form Feeds to make it easier to read and find things
{               Add quad word data type
{               Add NRP for sequential and relative files
{               Add a couple fields in XABCXR which are needed.
{
{       V03-008 JWH0139         Jeffrey W. Horn         29-Nov-1982
{               Add XAB$B_BIL, XAB$B_AIL, XAB$B_ATL to contain the
{               return lengths for journal names.
{               Change XAB$B_JOP XAB$W_JOP.
{               Add free space to end of journal XAB.
{
{       V03-007 KPL0002         Peter Lieberwirth       15-Nov-1982
{               Add XAB$C_MAXJNLNAM and XAB$K_MAXJNLNAM to define the
{               longest length of a journal name.
{
{       V03-006 LJA0033         Laurie J. Anderson       1-Nov-1982
{               Change around the NRP for ISAM slightly, and add couple
{               fields to XABCXF which are needed.
{
{       V03-005 MCN0008         Maria del C. Nasr       26-Oct-1982
{               KEY_NCMPR flag in the key XAB can be defined for all keys.
{
{       V03-004 LJA0028         Laurie J. Anderson      14-Oct-1982
{               Added a prologue version to be associated with the context XAB's
{               Add RMS Context Extraction NRP support for ISAM files
{               Took out a couple fields in the XABCXF not needed
{
```

```
{       V03-003 JWH0002         Jeffrey W. Horn         10-Sep-1982
{               Add $RAB ROP LV2 bit, to implement level 2 RU
{               locking consistancy.
{
{       V03-002 JWH0001         Jeffrey W. Horn         02-Jul-1982
{               Add XABJNL, journaling XAB.
{
{       V03-001 LJA0009         Laurie Anderson         09-Jun-1982
{               Add RAB$L_XAB, to support new context XAB (XABCXR)
{               Add two new XABs (of type context), one for FAB, other
{               for RAB, XABCXF and XABCXR, respectively.
{
{       V02-040 RAS0073         Ron Schaefer            2-Mar-1982
{               Add FAB$B_DSBMSK, to support $TRNLOG translation table
{               disable mask support.
{
{       V02-039 CDS0003         C Saether               5-Jan-1982
{               Add XAB$W_GBC and XAB$W_VERLIMIT to FHC XAB.
{               Make NAM$C_MAXRSS and NAM$C_MAXRSSLCL both 252.
{
{       V02-038 DMW0003         David Michael Walp      21-Jan-82
{               Remove NAM$B/L_QUOTED, 17 ANSI "a" character filenames
{
{       V02-037 KBT0001         Keith B Thompson        8-Jan-1982
{               Remove XAB$B_COMPAT, change XAB$B_STRUCT to XAB$B_PROLOG
{               in the key xab and add XAB$B_MTACC to the protection xab
{
{       V02-036 TMK0001         Todd M. Katz            8-Jan-1982
{               Define NAM$V_IFI and NAM$V_SRCHNMF in the field NAM$L_WCC.
{
{       V02-035 CDS0002         C Saether               23-Dec-1981
{               Add FAB$W_GBC field for global buffer count.
{
{       V02-034 LJA0002         Laurie Anderson         20-Dec-1981
{               Re-inserted NAM$C_BLN_DIRWC as equivalant to NAM$C_BLN
{
{       V02-033 CDS0001         C Saether               4-Nov-1981
{               Change key xab "structure" field to "struct".
{
{       V02-032 RAS0040         Ron Schaefer            26-Oct-1981
{               Add NAM$V_CNCL_DEV bit for concealed devices and
{               NAM$V_ROOT_DIR bit for rooted directories to the
{               NAM$L_FNB field.
{
{       V02-031 PSK0004         Paulina S Knibbe        19-Oct-1981
{               Change the XAB$$B_CMP_BITS to COMPAT and insert the
{               constants that field can take.
{
{       V02-030 PSK0003         Paulina S Knibbe        14-Sep-1981
{               Make the new key XAB variables shorter so we can
{               keep three column format in map
{
{       V02-029 PSK0002         Paulina S Knibbe        02-Sep-1981
{               Make the KEY XAB long word aligned again.
{
{       V02-028 PSK0001         Paulina S Knibbe        25-Aug-1981
```

```
RMS

mod
/**
/*
/*
/*
/**
/*
/*
/*
con

agg












/*-


end

end
```

```
{               Merge in Maria's changes to the KEY XAB.
{
{       V02-027 RAS0028         Ron Schaefer            20-Aug-1981
{               Change FAB$C_STM11 to FAB$C_STM.
{
{       V02-026 JAK0062         J A Krycka              14-Aug-1981
{               Add NOP and RFS fields to the NAM block.
{
{       V02-025 RAS0014         Ron Schaefer            7-Jul-1981
{               Add stream format codes to FAB and stream access code to RAB.
{
{       V02-024 RAS0012         Ron Schaefer            12-Jun-1981
{               Correct the BLISS definition of the XAB protection codes
{               to be relative to the start of the 4-bit protection field.
{
{       V02-023 JAK0059         J A Krycka              11-Jun-1981
{               Multiplex the QUOTED descriptor in the NAM block with the
{               NAME descriptor instead of the DEV descriptor.
{
{       V02-022 MCN0007         Maria del C. Nasr       12-May-1981
{               Use new symbol for old length of backup date and time XAB.
{
{       V02-021 KRM0012         Karl Malik              17-Apr-1981
{               Remove the NAM DWC definitions and extend the NAM block
{               by 40 bytes to provide easy access to various filespec
{               elements of either the expanded name string or the
{               resultant name string.
{
{       V02-020 MLJ0010         Martin L. Jack          25-Mar-1981
{               Add alternate format file ID in NAM block.
{
{       V02-019 kpl0001         Peter Lieberwirth       31-Dec-1980
{               Include definitions for new ROP bits RRL and REA.
{               Clean up some spelling and format while here.
{
{       V02-018 MCN0004         Maria del C. Nasr       17-Nov-1980
{               Include definition for backup date and time XAB.
{
```

```
RMS

mod
/*-
/*+
/*
/*
/*
/*
/*
/*+
/*
/*
con

agg




/*-
```

```
{
{       file access block (fab) definitions
{
{
module $FABDEF;

/+++++++++++
/*   the fields thru ctx must not be modified due to
/*   commonality between fab/rab/xab

aggregate FABDEF structure prefix FAB$;
    BID byte unsigned;                                  /* block id
    constant BID        equals 3  prefix FAB tag $C;    /* code for fab
    BLN byte unsigned;                                  /* block len
    IFI_OVERLAY union;
        IFI word unsigned;                              /* internal file index
        IFI_BITS structure;
            FILL_1 bitfield length 6 fill prefix FABDEF tag $$;/* move to bit 6
            PPF_RAT bitfield mask length 8;             /* rat value for process-permanent files
            PPF_IND bitfield mask;                      /* indirect access to process-permanent file
                                                        /* (i.e., restricted operations)
        end IFI_BITS;
    end IFI_OVERLAY;
    FOP_OVERLAY union;
        FOP longword unsigned;                          /* file options
        FOP_BITS structure;
            FILL_2 bitfield fill prefix FABDEF tag $$;  /* reserved for asy (not implemented)
            MXV bitfield mask;                          /* maximize version number
            SUP bitfield mask;                          /* supersede existing file
            TMP bitfield mask;                          /* create temporary file
            TMD bitfield mask;                          /* create temp file marked for delete
            DFW bitfield mask;                          /* deferred write (rel and idx)
            SQO bitfield mask;                          /* sequential access only
            RWO bitfield mask;                          /* rewind mt on open
            POS bitfield mask;                          /* use next magtape position
            WCK bitfield mask;                          /* write checking
            NEF bitfield mask;                          /* inhibit end of file positioning
            RWC bitfield mask;                          /* rewind mt on close
            DMO bitfield mask;                          /* dismount mt on close (not implemented)
            SPL bitfield mask;                          /* spool file on close
            SCF bitfield mask;                          /* submit command file on close
            DLT bitfield mask;                          /* delete sub-option
            NFS bitfield mask;                          /* non-file structured operation
            UFO bitfield mask;                          /* user file open - no rms operations
            PPF bitfield mask;                          /* process permanent file (pio segment)
            INP bitfield mask;                          /* process-permanent file is 'input'
            CTG bitfield mask;                          /* contiguous extension
            CBT bitfield mask;                          /* contiguous best try
            FILL_3 bitfield fill prefix FABDEF tag $$;  /* reserved (not implemented)
            RCK bitfield mask;                          /* read checking
            NAM bitfield mask;                          /* use name block dvi, did, and/or fid fields for open
            CIF bitfield mask;                          /* create if non-existent
            FILL_4 bitfield fill prefix FABDEF tag $$;  /* reserved (was UFM bitfield)
            ESC bitfield mask;                          /* 'escape' to non-standard function ($modify)
            TEF bitfield mask;                          /* truncate at eof on close (write-accessed seq. disk file only)
            OFP bitfield mask;                          /* output file parse (only name type sticky)
            KFO bitfield mask;                          /* known file open (image activator only release 1)
            FILL_5 bitfield fill prefix FABDEF tag $$;  /* reserved (not implemented)
        end FOP_BITS;
    end FOP_OVERLAY;
    STS longword unsigned;                              /* status
    STV longword unsigned;                              /* status value
    ALQ longword unsigned;                              /* allocation quantity
    DEQ word unsigned;                                  /* default allocation quantity
    FAC_OVERLAY union;
        FAC byte unsigned;                              /* file access
        FAC_BITS structure;
            PUT bitfield mask;                          /* put access
            GET bitfield mask;                          /* get access
            DEL bitfield mask;                          /* delete access
            UPD bitfield mask;                          /* update access
            TRN bitfield mask;                          /* truncate access
            BIO bitfield mask;                          /* block i/o access
            BRO bitfield mask;                          /* block and record i/o access
            EXE bitfield mask;                          /* execute access (caller must be exec or kernel mode,
                                                        /*  ufo must also be set)
        end FAC_BITS;
    end FAC_OVERLAY;
    SHR_OVERLAY union;
        SHR byte unsigned;                              /* file sharing
        SHR_BITS structure;
            SHRPUT bitfield mask;                       /* put access
            SHRGET bitfield mask;                       /* get access
            SHRDEL bitfield mask;                       /* delete access
            SHRUPD bitfield mask;                       /* update access
            MSE bitfield mask;                          /* multi-stream connects enabled
            NIL bitfield mask;                          /* no sharing
            UPI bitfield mask;                          /* user provided interlocking (allows multiple
                                                        /*  writers to seq. files)
        end SHR_BITS;
    end SHR_OVERLAY;
    CTX longword unsigned;                              /* user context
/*-----*****
    RTV byte;                                           /* retrieval window size
    ORG_OVERLAY union;
        ORG byte unsigned;                              /* file organization
        ORG_BITS structure;
            FILL_6 bitfield length 4 fill prefix FABDEF tag $$;
            ORG bitfield length 4;
        end ORG_BITS;
        constant SEQ    equals 0  prefix FAB tag $C;    /* sequential
        constant REL    equals 16  prefix FAB tag $C;   /* relative
        constant IDX    equals 32  prefix FAB tag $C;   /* indexed
        constant HSH    equals 48  prefix FAB tag $C;   /* hashed
    end ORG_OVERLAY;
    RAT_OVERLAY union;
        RAT byte unsigned;                              /* record format
        RAT_BITS structure;
            FTN bitfield mask;                          /* fortran carriage-ctl
            CR bitfield mask;                           /* lf-record-cr carriage ctl
            PRN bitfield mask;                          /* print-file carriage ctl
            BLK bitfield mask;                          /* records don't cross block boundaries
        end RAT_BITS;
    end RAT_OVERLAY;
    RFM byte unsigned;                                          /* record format
    constant RFM_DFLT   equals 2  prefix FAB tag $C;            /* var len is default
    constant UDF        equals 0  prefix FAB tag $C;            /* undefined (also stream binary)
    constant FIX        equals 1  orefix FAB tag $C;            /* fixed length records
    constant VAR        equals 2  prefix FAB tag $C;            /* variable length records
    constant VFC        equals 3  prefix FAB tag $C;            /* variable fixed control
    constant STM        equals 4  prefix FAB tag $C;            /* RMS-11 stream (valid only for sequential org)
    constant STMLF      equals 5  prefix FAB tag $C;            /* LF stream (valid only for sequential org)
    constant STMCR      equals 6  prefix FAB tag $C;            /* CR stream (valid only for sequential org)
    constant MAXRFM     equals 6  prefix FAB tag $C;            /* maximum rfm supported
    JNL longword unsigned;                                      /* lcb address
    XAB longword unsigned;                                      /* xab address
    NAM longword unsigned;                                      /* nam block address
    FNA longword unsigned;                                      /* file name string address
    DNA longword unsigned;                                      /* default file name string addr
    FNS byte unsigned;                                          /* file name string size
    DNS byte unsigned;                                          /* default name string size
    MRS word unsigned;                                          /* maximum record size
    MRN longword unsigned;                                      /* maximum record number
    BLS word unsigned;                                          /* blocksize for tape
    BKS byte unsigned;                                          /* bucket size
    FSZ byte unsigned;                                          /* fixed header size
    DEV longword unsigned;                                      /* device characteristics
    SDC longword unsigned;                                      /* spooling device characteristics
    GBC word unsigned;                                          /* Global buffer count
    ACMODES_OVERLAY union;
        ACMODES byte unsigned;                                  /* agent access modes
        ACMODES_BITS structure;
            LNM_MODE bitfield length 2;                         /* ACMODE for log nams
            CHAN_MODE bitfield length 2;                        /* ACMODE for channel
            FILE_MODE bitfield length 2;                        /* ACMODE to use for determining file accessibility
            FILL_7 bitfield length 2 fill prefix FABDEF tag $$;
        end ACMODES_BITS;
    end ACMODES_OVERLAY;
    RCF_OVERLAY union;                                          /* recovery control flags
        RCF byte unsigned;
        RCF_BITS structure;
            RU bitfield mask;                                   /* recovery unit recovery
            AI bitfield mask;                                   /* after image recovery
            BI bitfield mask;                                   /* before image recovery
        end RCF_BITS;
    end RCF_OVERLAY;
    FILL_8 longword fill prefix FABDEF tag $$;                  /* (spare)
    constant BLN equals . prefix FAB$ tag K;                    /* length of fab
    constant BLN equals . prefix FAB$ tag C;                    /* length of fab
end FABDEF;

end_module $FABDEF;
```

```
module $RABDEF;
/*
/*         record access block (rab) definitions
/*
/*  there is one rab per connected stream
/*  it is used for all communications between the user
/*  and rms concerning operations on the stream
/*

/*++++++++++++
/*  the fields thru ctx cannot be changed due to commonality
/*  with the fab
/*

aggregate RABDEF structure prefix RAB$;
    BID byte unsigned;                               /* block id
    constant BID        equals 1  prefix RAB tag $C;    /* code for rab
    BLN byte unsigned;                               /* block length
    ISI_OVERLAY union;
        ISI word unsigned;                           /* internal stream index
                                                     /* (ifi in fab)

        ISI_BITS structure;
            FILL_1 bitfield length 6 fill prefix RABDEF tag $$;/* move to bit 6
            PPF_RAT bitfield mask length 8;          /* rat value for process-permanent files
            PPF_IND bitfield mask;                   /* indirect access to process-permanent file
                                                     /* (i.e., restricted operations)
        end ISI_BITS;
    end ISI_OVERLAY;
    ROP_OVERLAY union;
        ROP longword unsigned;                       /* record options
        ROP_BITS0 structure;
            ASY bitfield mask;                       /* asynchronous operations
            TPT bitfield mask;                       /* truncate put - allow sequential put not at
                                                     /*   eof, thus truncating file (seq. org only)
                                                     /*
                                                     /* these next two should be in the byte for bits
                                                     /* input to $find or $get, but there is no room there
                                                     /*
            REA bitfield mask;                       /* lock record for read only, allow other readers
            RRL bitfield mask;                       /* read record regardless of lock
                                                     /*
            UIF bitfield mask;                       /* update if existent
            MAS bitfield mask;                       /* mass-insert mode
            FDL bitfield mask;                       /* fast record deletion
            HSH bitfield mask;                       /* use hash code in bkt
                                                     /*
            EOF bitfield mask;                       /* connect to eof
            RAH bitfield mask;                       /* read ahead
            WBH bitfield mask;                       /* write behind
            BIO bitfield mask;                       /* connect for bio only
            LV2 bitfield mask;                       /* level 2 RU lock consistancy
            LOA bitfield mask;                       /* use bucket fill percentage
            LIM bitfield mask;                       /* compare for key limit reached on $get/$find seq. (idx only)
            FILL_2 bitfield fill prefix RABDEF tag $$;  /* (1 spare)
                                                     /*
                                                     /* the following bits are input to
```

```
                                                           /* $find or $get, (see above also REA and RRL)
                                                           /* (separate byte)
                                                           /*
            LOC bitfield mask;                             /* use locate mode
            WAT bitfield mask;                             /* wait if record not available
            ULK bitfield mask;                             /* manual unlocking
            RLK bitfield mask;                             /* allow readers for this locked record
            NLK bitfield mask;                             /* do not lock record
            KGE bitfield mask;                             /* key > or =
            KGT bitfield mask;                             /* key greater than
            NXR bitfield mask;                             /* get non-existent record
                                                           /*
                                                           /*  the following bits are terminal qualifiers only
                                                           /*  (separate byte)
                                                           /*
            RNE bitfield mask;                             /* read no echo
            TMO bitfield mask;                             /* use time-out period
            CVT bitfield mask;                             /* convert to upper case
            RNF bitfield mask;                             /* read no filter
            ETO bitfield mask;                             /* extended terminal operation
            PTA bitfield mask;                             /* purge type ahead
            PMT bitfield mask;                             /* use prompt buffer
            CCO bitfield mask;                             /* cancel control o on output
        end ROP_BITS0;

                                                           /* the following bits may be
                                                           /* input to various rab-related
                                                           /* operations
                                                           /*
        ROP_FIELDS structure;
            FILL_3 byte fill prefix RABDEF tag $$;
            ROP1 byte unsigned;                            /* various options
            ROP2 byte unsigned;                            /* get/find options (use of this field discouraged
                                                           /* due to REA and RRL being in a different byte)
            ROP3 byte unsigned;                            /* terminal read options
/*
        end ROP_FIELDS;
    end ROP_OVERLAY;
    STS longword unsigned;                                 /* status
    STV_OVERLAY union;
        STV longword unsigned;                             /* status value
        STV_FIELDS structure;
            STV0 word unsigned;                            /* low word of stv
            STV2 word unsigned;                            /* high word of stv
        end STV_FIELDS;
    end STV_OVERLAY;
    RFA_OVERLAY union;
        RFA word unsigned dimension 3;                     /* record's file address
        RFA_FIELDS structure;
            RFA0 longword unsigned;
            RFA4 word unsigned;
        end RFA_FIELDS;
    end RFA_OVERLAY;
    FILL_4 word fill prefix RABDEF tag $$;                 /* (reserved - rms release 1 optimizes stores
                                                           /*  to the rfa field to be a move quad, overwriting
                                                           /*  this reserved word)
    CTX longword unsigned;                                  /* user context
/*-----*****
    FILL_5 word fill prefix RABDEF tag $$;                  /* (spare)
    RAC byte unsigned;                                      /* record access
    constant SEQ        equals 0  prefix RAB tag $C;        /* sequential access
    constant KEY        equals 1  prefix RAB tag $C;        /* keyed access
    constant RFA        equals 2  prefix RAB tag $C;        /* rfa access
    constant STM        equals 3  prefix RAB tag $C;        /* stream access (valid only for sequential org)
    TMO byte unsigned;                                      /* time-out period
    USZ word unsigned;                                      /* user buffer size
    RSZ word unsigned;                                      /* record buffer size
    UBF longword unsigned;                                  /* user buffer address
    RBF longword unsigned;                                  /* record buffer address
    RHB longword unsigned;                                  /* record header buffer addr
    KBF_OVERLAY union;
        KBF longword unsigned;                              /* key buffer address
        PBF longword unsigned;                              /* prompt buffer addr
    end KBF_OVERLAY;
    KSZ_OVERLAY union;
        KSZ byte unsigned;                                  /* key buffer size
        PSZ byte unsigned;                                  /* prompt buffer size
    end KSZ_OVERLAY;
    KRF byte unsigned;                                      /* key of reference
    MBF byte;                                               /* multi-buffer count
    MBC byte unsigned;                                      /* multi-block count
    BKT_OVERLAY union;
        BKT longword unsigned;                              /* bucket hash code, vbn, or rrn
        DCT longword unsigned;                              /* duplicates count on key accessed on alternate key
    end BKT_OVERLAY;
    FAB longword unsigned;                                  /* related fab for connect
    XAB longword unsigned;                                  /* XAB address
    constant BLN equals . prefix RAB$ tag K;                /* length of rab
    constant BLN equals . prefix RAB$ tag C;                /* length of rab
end RABDEF;

end_module $RABDEF;
```

```
module $NAMDEF;
/*
/*          name block field definitions
/*
/*  the nam block is used to communicate optional
/*  filename-related information
/*


aggregate NAMDEF structure prefix NAM$;
    BID byte unsigned;                                      /* block id
    constant BID          equals 2  prefix NAM tag $C;     /* code for nam block
    BLN byte unsigned;                                      /* block length
/*+++++++++++++++++++++++
/*  the following 3 fields must not be rearranged relative to each other
/*
    RSS byte unsigned;                                      /* resultant string area size
    RSL byte unsigned;                                      /* resultant string length
    RSA longword unsigned;                                  /* resultant string area address
/*----------------------

    constant MAXRSS      equals 255  prefix NAM tag $C;   /* maximum resultant name string size (network)
    constant MAXRSSLCL   equals 255  prefix NAM tag $C;   /* maximum resultant name string size (local)

    NOP_OVERLAY union;
        NOP byte unsigned;                                  /* Name options
        NOP_BITS structure;
            PWD bitfield mask;                              /* Return password if present in nodespec string and any
                                                            /* other task-specific data of the form /netacp_data"
                                                            /*  (default is to mask out password from expanded and
                                                            /*  resultant name strings and to create a logical name
                                                            /*  whose equivalence string is the unaltered nodespec)
            FILL_1 bitfield mask;                           /* unused.  (used to be undocumented ROD)
            FILL_2 bitfield mask;                           /* unused.  (used to be undocumented SOD)
            SYNCHK bitfield mask;                           /* Only do syntax check on $parse operation
            NOCONCEAL bitfield mask;                        /* Do not conceal device/root directory
            SLPARSE bitfield mask;                          /* Parse search list (not documented) -- used by BACKUP.
            SRCHXABS bitfield mask;                         /* Fill in attached XABS on $SEARCH operations over the
                                                            /*  network (not documented) -- used by directory.
        end NOP_BITS;
    end NOP_OVERLAY;
    RFS byte unsigned;                                      /* Remote file system type (currently not documented)
                                                            /* Note: This field is reserved for use by Digital
    constant UFS          equals 0  prefix NAM tag $C;     /* Unknown file system for remote file access or
                                                            /*  not applicable for local file access or
                                                            /*  not applicable for task-to-task operation
    constant RMS11        equals 1  prefix NAM tag $C;     /* RMS-11
    constant RMS20        equals 2  prefix NAM tag $C;     /* RMS-20
    constant RMS32        equals 3  prefix NAM tag $C;     /* RMS-32
    constant FCS11        equals 4  prefix NAM tag $C;     /* FCS-11
    constant RT11FS       equals 5  prefix NAM tag $C;     /* RT-11 file system
    constant TOPS20FS     equals 7  prefix NAM tag $C;     /* TOPS-20 file system
    constant TOPS10FS     equals 8  prefix NAM tag $C;     /* TOPS-10 file system
    constant RMS32S       equals 10  prefix NAM tag $C;    /* RMS-32 subset (e.g., VAXELAN)
/*+++++++++++++++++++++++
/*  the following 3 fields must not be rearranged relative to each other
```

```
/*
    ESS byte unsigned;                                      /* espanded string area size
    ESL byte unsigned;                                      /* expanded string length
    ESA longword unsigned;                                  /* expanded string area address
/*-----------------------
    RLF longword unsigned;                                  /* related file nam block addr
    DVI character length 16;                                /* device id
    constant DVI          equals 16  prefix NAM tag $C;     /* length of dvi field
/*+++++++++++
/*  the location of the following fields must not
/*  be changed due to their commonality with the fib
    FID_OVERLAY union;
        FID word unsigned dimension 3;                      /* file id
        FID_FIELDS structure;
            FID_NUM word unsigned;                          /* file number
            FID_SEQ word unsigned;                          /* sequence number
            FID_RVN_OVERLAY union;
                FID_RVN word unsigned;                      /* relative volume number
                FID_RVN_FIELDS structure;
                    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;
    DID_OVERLAY union;
        DID word unsigned dimension 3;                      /* directory id
        DID_FIELDS structure;
            DID_NUM word unsigned;                          /* file number
            DID_SEQ word unsigned;                          /* sequence number
            DID_RVN_OVERLAY union;
                DID_RVN word unsigned;                      /* relative volume number
                DID_RVN_FIELDS structure;
                    DID_RVN byte unsigned;                  /* alternate format RVN
                    DID_NMX byte unsigned;                  /* alternate format file number extension
                end DID_RVN_FIELDS;
            end DID_RVN_OVERLAY;
        end DID_FIELDS;
    end DID_OVERLAY;
    WCC_OVERLAY union;
        WCC longword unsigned;                              /* wild card context
        WCC_BITS structure;
            FILL_1 bitfield length 16 fill prefix NAMDEF tag $$;/* the first word is reserved for IFI/ACP context
            IFI bitfield mask;                              /* the first word contains an IFI
            FILL_2 bitfield length 13 fill prefix NAMDEF tag $$;/* grow from top down, start at top bit
            SRCHNMF bitfield mask;                          /* no-more-files has been encountered on a search
            SVCTX bitfield mask;                            /* save context across search calls
        end WCC_BITS;
    end WCC_OVERLAY;
    FNB_OVERLAY union;
        FNB longword unsigned;                              /* file name status bits
        constant BLN_V2 equals . prefix NAM$ tag K;         /* Version 2 name block length
        constant BLN_V2 equals . prefix NAM$ tag C;         /* Version 2 name block length
        FNB_BITS0 structure;
            EXP_VER bitfield mask;                          /* version was explicit
            EXP_TYPE bitfield mask;                         /* type was explicit
            EXP_NAME bitfield mask;                             /* name was explicit
            WILD_VER bitfield mask;                             /* version contained a wild card
            WILD_TYPE bitfield mask;                            /* type contained a wild card
            WILD_NAME bitfield mask;                            /* name contained a wild card
            EXP_DIR bitfield mask;                              /* directory was explicit
            EXP_DEV bitfield mask;                              /* device was explicit
            WILDCARD bitfield mask;                             /* filename string included a wild card
                                                                /* (inclusive or of other wild card bits)
            FILL_3 bitfield length 2 fill prefix NAMDEF tag $$;/* (spares)
            SEARCH_LIST bitfield mask;                          /* search list present
            CNCL_DEV bitfield mask;                             /* concealed device present
            ROOT_DIR bitfield mask;                             /* root directory present
            LOWVER bitfield mask;                               /* lower numbered version(s) of file exist(s)
            HIGHVER bitfield mask;                              /* higher "
                                                                /*
            PPF bitfield mask;                                  /* process-permanent file referenced indirectly
            NODE bitfield mask;                                 /* filename specification included a nodename
            QUOTED bitfield mask;                               /* filename spec included a quoted string
            GRP_MBR bitfield mask;                              /* directory spec was of group-member format
            WILD_DIR bitfield mask;                             /* directory spec included a wild card
            DIR_LVLS bitfield mask length 3;                    /* number of directory levels (0=ufd only)
        end FNB_BITS0;
        FNB_BITS1 structure;
            FILL_4 bitfield length 24 fill prefix NAMDEF tag $$;/* separate byte for wild card directory flags
            WILD_UFD bitfield mask;                             /* ufd included a wild card
            WILD_SFD1 bitfield mask;                            /* sfd1 included a wild card
            WILD_SFD2 bitfield mask;                            /* sfd2 included a wild card
            WILD_SFD3 bitfield mask;                            /* sfd3 included a wild card
            WILD_SFD4 bitfield mask;                            /* sfd4 included a wild card
            WILD_SFD5 bitfield mask;                            /* sfd5 included a wild card
            WILD_SFD6 bitfield mask;                            /* sfd6 included a wild card
            WILD_SFD7 bitfield mask;                            /* sfd7 included a wild card
        end FNB_BITS1;
        FNB_BITS2 structure;
            FILL_5 bitfield length 24 fill prefix NAMDEF tag $$;/* alternate definitions for wild_ufd and wild_sfd1
            WILD_GRP bitfield mask;                             /* group contained a wild card
            WILD_MBR bitfield mask;                             /* member contained a wild card
        end FNB_BITS2;
/*-----*****
                                                                /* (prior to 40 byte extension)
                                                                                                                                /*
/*                                                                                                                              /*
/* Extend the NAM block by 40 bytes.                                                                                            /*
/*                                                                                                                              /*
    end FNB_OVERLAY;
    NODE byte unsigned;                                         /* Nodespec length
    DEV byte unsigned;                                          /* Device length
    DIR byte unsigned;                                          /* Directory length
    NAME byte unsigned;                                         /* Filename length                                          /*
    TYPE byte unsigned;                                         /* Filetype length                                          /*
    VER byte unsigned;                                          /* Version number length                                    /*
    FILL_6 byte dimension 2 fill prefix NAMDEF tag $$;          /* Currently unused
    NODE longword unsigned;                                     /* Nodespec address
    DEV longword unsigned;                                      /* Device address
    DIR longword unsigned;                                      /* Directory address
    NAME longword unsigned;                                     /* Filename address
    TYPE longword unsigned;                                   /* Filetype address
    VER longword unsigned;                                    /* Version number address
    FILL_7 longword dimension 2 fill prefix NAMDEF tag $$;/* Currently unused
    constant BLN_DIRWC equals . prefix NAM$ tag K;           /* Not documented optional length
    constant BLN_DIRWC equals . prefix NAM$ tag C;           /* Not documented optional length
    constant BLN equals . prefix NAM$ tag K;                 /* Name block length
    constant BLN equals . prefix NAM$ tag C;                 /* Name block length
end NAMDEF;

end_module $NAMDEF;
```

RMS

```
/*-
/*+
end

end
```

```
module $XABDEF;
/*
/*          definitions for all xabs
/*                 $xabdef
/*
/*
/*
/*  the first four fields are shared in common between all xabs
/*  and hence are defined only once
/*  (the only exception is that the spare word may be used by some xabs)
/*

aggregate XABDEF structure prefix XAB$;
    COD byte unsigned;                                  /* xab id code
    BLN byte unsigned;                                  /* block length
    FILL_1 word fill prefix XABDEF tag $$;              /* (spare)
    NXT longword unsigned;                              /* xab chain link
                                                        /*WITH POSSIBLE EXCEPTION OF SPARE FIELD
    RVN word unsigned;
    FILL_2 word fill prefix XABDEF tag $$;
    RDT_OVERLAY union;
        RDT quadword;
        RDT_FIELDS structure;
            RDT0 longword unsigned;
            RDT4 longword;
                                                        /*COMMON AMONG DAT AND RDT XABS
        end RDT_FIELDS;
    end RDT_OVERLAY;
end XABDEF;

aggregate XABDEF1 structure prefix XAB$;
    FILL_3 byte dimension 8 fill prefix XABDEF tag $$;
    FILL_4 byte fill prefix XABDEF tag $$;              /*THESE FIELDS WILL BE DEFINED LATER
    FILL_5 byte fill prefix XABDEF tag $$;
    FILL_6 word fill prefix XABDEF tag $$;
    FILL_7 longword fill prefix XABDEF tag $$;
    FILL_8 longword fill prefix XABDEF tag $$;
    FILL_9 word fill prefix XABDEF tag $$;
    BKZ byte unsigned;                                  /*COMMON TO FHC AND ALQ XABS

end XABDEF1;

constant CXT_VER1 equals 1  prefix XAB tag $C;          /* RMS Context Extraction version 1

end_module $XABDEF;
```

```
module $XABFHCDEF;
/***
/*       file header characteristics xab definitions
/*              $xabfhcdef
/*
/***********
/*  the fields of this xab cannot be rearranged since
/*  they correspond to an cn-disk structure
/*
constant FHC    equals 29  prefix XAB tag $C;          /* xabfhc id code

aggregate XABFHCDEF structure prefix XAB$;
    FILL_1 byte fill prefix XABFHCDEF tag $$;
    FILL_2 byte fill prefix XABFHCDEF tag $$;
    FILL_3 word fill prefix XABFHCDEF tag $$;
    FILL_4 longword fill prefix XABFHCDEF tag $$;      /*HAS SAME COD, BLN, SPARE AND NXT FIELD
                                                       /*THESE 4 FIELDS ARE COMMON TO ALL XABS AND
                                                       /*HAVE BEEN DEFINED BY $XABDEF
    RFO byte unsigned;                                 /* record format and file org
    ATR byte unsigned;                                 /* record attributes
    LRL word unsigned;                                 /* longest record's length
    HBK_OVERLAY union;
        HBK longword unsigned;                         /* hi vbn allocated
                                                       /* (n.b. reversed on disk!)
        HBK_FIELDS structure;
            HBK0 word unsigned;
            HBK2 word unsigned;
        end HBK_FIELDS;
    end HBK_OVERLAY;
    EBK_OVERLAY union;
        EBK longword unsigned;                         /* eof vbn
                                                       /* (n.b. reversed on disk)
        EBK_FIELDS structure;
            EBK0 word unsigned;
            EBK2 word unsigned;
        end EBK_FIELDS;
    end EBK_OVERLAY;
    FFB word unsigned;                                 /* first free byte in eof block
    FILL_5 byte fill prefix XABFHCDEF tag $$;          /* bucket size for fhc ( note: field name is bkz,
                                                       /* defined above in $xabdef, since it is shared
                                                       /* by the all xab)
    HSZ byte unsigned;                                 /* header size for vfc
    MRZ word unsigned;                                 /* max record size
    DXQ word unsigned;                                 /* default extend quantity
    GBC word unsigned;                                 /* global buffer count
    FILL_6 byte dimension 8 fill prefix XABFHCDEF tag $$;/* spares (pad to last word)
    VERLIMIT word unsigned;                            /* version limit for file.
/*-----*****
    SBN longword unsigned;                             /* starting lbn if contiguous
    constant FHCLEN equals . prefix XAB$ tag K;        /* length of xabfhc
    constant FHCLEN equals . prefix XAB$ tag C;        /* length of xabfhc
end XABFHCDEF;

end_module $XABFHCDEF;
```

```
module $XABALLDEF;
/*--
/*++
/*
/*        allocation xab definitions
/*                $xaballdef
/*
/*
/*++++++++++++
/*  the fields thru bkz cannot be rearranged due to
/*  their commonality with fab
constant ALL    equals 20  prefix XAB tag $C;           /* xaball id code

aggregate XABALLDEF structure prefix XAB$;
    FILL_1 byte fill prefix XABALLDEF tag $$;
    FILL_2 byte fill prefix XABALLDEF tag $$;
    FILL_3 word fill prefix XABALLDEF tag $$;
    FILL_4 longword fill prefix XABALLDEF tag $$;       /*HAS SAME COD, BLN, SPARE AND NXT FIELD
                                                        /*THESE 4 FIELDS ARE COMMON TO ALL XABS AND
                                                        /*HAVE BEEN DEFINED BY $XABDEF
    AOP_OVERLAY union;
        AOP byte unsigned;                              /* allocation options
        AOP_BITS structure;
            HRD bitfield mask;                          /* fail if requested alignment impossible
            ONC bitfield mask;                          /* locate allocated space within a cylinder
            FILL_5 bitfield length 3 fill prefix XABALLDEF tag $$;/* (spares)
            CBT bitfield mask;                          /* contiguous alllocation, best try
            FILL_6 bitfield fill prefix XABALLDEF tag $$;/* spare
            CTG bitfield mask;                          /* contiguous allocation
        end AOP_BITS;
    end AOP_OVERLAY;
    ALN byte unsigned;                                  /* alignment type
    constant "ANY"      equals 0  prefix XAB tag $C;    /* any allocation o.k.
    constant CYL        equals 1  prefix XAB tag $C;    /* cylinder boundary
    constant LBN        equals 2  prefix XAB tag $C;    /* allocate at specified lbn
    constant VBN        equals 3  prefix XAB tag $C;    /* allocate near specified vbn
    constant RFI        equals 4  prefix XAB tag $C;    /* allocate near related file
    VOL word unsigned;                                  /* relative volume no. for allocation
                                                        /* (not applicable if aln = vbn or rfi)
    LOC longword unsigned;                              /* allocation location
    ALQ longword unsigned;                              /* allocation quantity
    DEQ word unsigned;                                  /* default allocation quantity
    FILL_7 byte fill prefix XABALLDEF tag $$;           /* bucket size for area (note: field name is bkz,
                                                        /* defined above in $xabdef,since it is shared by the fhc
                                                        /* xab and has the same offset, of course)
/*-----*****
    AID byte unsigned;                                  /* area id number
    RFI_OVERLAY union;
        RFI word unsigned dimension 3;                  /* related file id
        RFI_FIELDS structure;
            RFI0 word unsigned;                         /* file number
            RFI2 word unsigned;                         /* seq number
            RFI4 word unsigned;                         /* rev number
        end RFI_FIELDS;
    end RFI_OVERLAY;
    FILL_8 word fill prefix XABALLDEF tag $$;           /* (spare)
    constant ALLLEN equals . prefix XAB$ tag K;         /* length of xaball
    constant ALLLEN equals . prefix XAB$ tag C;         /* length of xaball
end XABALLDEF;

end_module $XABALLDEF;
```

```
/*
/*
/*

end

end
```

```
module $XABDATDEF;
/*--
/*++
/*
/*       date/time xab definitions
/*               $xabdatdef
/*
constant DAT    equals 18  prefix XAB tag $C;          /* xabdat id code

aggregate XABDATDEF structure prefix XAB$;
    FILL_1 byte fill prefix XABDATDEF tag $$;
    FILL_2 byte fill prefix XABDATDEF tag $$;
    FILL_3 word fill prefix XABDATDEF tag $$;
    FILL_4 longword fill prefix XABDATDEF tag $$;       /*HAS SAME COD, BLN, SPARE AND NXT FIELD
                                                        /*THESE 4 FIELDS ARE COMMON TO ALL XABS AND
                                                        /*HAVE BEEN DEFINED BY $XABDEF
    FILL_5 word fill prefix XABDATDEF tag $$;           /*REVISION !,DEFINED IN $XABDEF,SINCE COMMON TO DAT & RDT
    FILL_6 word fill prefix XABDATDEF tag $$;           /* spare
    FILL_7 quadword fill prefix XABDATDEF tag $$;       /* revision date & time,defined in $xabdef
    CDT_OVERLAY union;
        CDT quadword;                                   /* creation date & time
        CDT_FIELDS structure;
            CDT0 longword unsigned;
            CDT4 longword;
        end CDT_FIELDS;
    end CDT_OVERLAY;
    EDT_OVERLAY union;
        EDT quadword;                                   /* expiration date & time
        constant DATLEN_V2 equals . prefix XAB$ tag K;  /* Version 2 XABDAT length
        constant DATLEN_V2 equals . prefix XAB$ tag C;  /* Version 2 XABDAT length
        EDT_FIELDS structure;
            EDT0 longword unsigned;
            EDT4 longword;
        end EDT_FIELDS;
    end EDT_OVERLAY;
    BDT_OVERLAY union;
        BDT quadword;                                   /* backup date and time
        constant DATLEN equals . prefix XAB$ tag K;     /* length of XABDAT
        constant DATLEN equals . prefix XAB$ tag C;     /* length of XABDAT
        BDT_FIELDS structure;
            BDT0 longword unsigned;
            BDT4 longword;
        end BDT_FIELDS;
    end BDT_OVERLAY;
end XABDATDEF;

end_module $XABDATDEF;
```

```
module $XABRDTDEF;
/*--
/*++
/*
/*          revision date/time xab definitions
/*                   $xabrdtdef
constant RDT    equals 30  prefix XAB tag $C;           /* xabrdt id code

aggregate XABRDTDEF structure prefix XAB$;
    FILL_1 byte fill prefix XABRDTDEF tag $$;
    FILL_2 byte fill prefix XABRDTDEF tag $$;
    FILL_3 word fill prefix XABRDTDEF tag $$;
    FILL_4 longword fill prefix XABRDTDEF tag $$;       /*HAS SAME COD, BLN, SPARE AND NXT FIELD
                                                        /*THESE 4 FIELDS ARE COMMON TO ALL XABS AND
                                                        /*HAVE BEEN DEFINED BY $XABDEF
    FILL_5 word fill prefix XABRDTDEF tag $$;           /*REVISION !,DEFINED IN $XABDEF,SINCE COMMON TO DAT & RDT
    FILL_6 word fill prefix XABRDTDEF tag $$;           /* spare
    FILL_7 quadword fill prefix XABRDTDEF tag $$;       /* revision date & time,defined in $xabdef
    constant RDTLEN equals . prefix XAB$ tag K;         /* length of rdt xab
    constant RDTLEN equals . prefix XAB$ tag C;         /* length of rdt xab
end XABRDTDEF;

end_module $XABRDTDEF;
```

```
module $XABPRODEF;
/*--
/*++
/*
/*       protection xab field definitions
/*               $xabprodef
/*
/*
constant PRO    equals 19  prefix XAB tag $C;           /* xabpro id code

aggregate XABPRODEF  union prefix XAB$;
    XABPRODEF_BITS structure;
        NOREAD bitfield mask;                           /* deny read access
        NOWRITE bitfield mask;                          /* deny write access
        NOEXE bitfield mask;                            /* deny execution access
        NODEL bitfield mask;                            /* deny delete access
    end XABPRODEF_BITS;
end XABPRODEF;

    aggregate XABPRODEF1 structure prefix XAB$;
    FILL_1 byte fill prefix XABPRODEF tag $$;
    FILL_2 byte fill prefix XABPRODEF tag $$;
    FILL_3 word fill prefix XABPRODEF tag $$;
    FILL_4 longword fill prefix XABPRODEF tag $$;       /*HAS SAME COD, BLN, SPARE AND NXT FIELD
                                                        /*THESE 4 FIELDS ARE COMMON TO ALL XABS AND
                                                        /*HAVE BEEN DEFINED BY $XABDEF
    PRO_OVERLAY union;
        PRO word unsigned;                              /* protection mask
        PRO_BITS structure;
            SYS bitfield length 4;                      /* system
            OWN bitfield length 4;                      /* owner
            GRP bitfield length 4;                      /* group
            WLD bitfield length 4;                      /* world
        end PRO_BITS;
    end PRO_OVERLAY;
    MTACC byte unsigned;                                /* Magtape access control char.
    PROT_OPT_OVERLAY union;
        PROT_OPT byte unsigned;                         /* XABPRO options field
        PROT_OPT_FIELDS structure;
            PROPAGATE bitfield mask;                    /* Propagate security attributes on $ENTER and $RENAME
        end PROT_OPT_FIELDS;
    end PROT_OPT_OVERLAY;
    UIC_OVERLAY union;
        UIC longword unsigned;                          /* uic code
        constant PROLEN_V3 equals . prefix XAB$ tag K;  /* V3a xabpro length
        constant PROLEN_V3 equals . prefix XAB$ tag C;  /* V3a xabpro length
        UIC_FIELDS structure;
            MBM word unsigned;                          /* member code
            GRP word unsigned;                          /* group code
        end UIC_FIELDS;
    end UIC_OVERLAY;

    PROT_MODE_OVERLAY union;                            /* RWED/mode protection for file
        PROT_MODE quadword;                             /* eventually may be a quadword
        PROT_MODE_FIELDS structure;
            PROT_MODE byte unsigned;                    /* but currently only a byte
        end PROT_MODE_FIELDS;
    end PROT_MODE_OVERLAY;

    ACLBUF longword unsigned;                          /* address of user's ACL buffer
    ACLSIZ word unsigned;                              /* size of user's ACL buffer
    ACLLEN word unsigned;                              /* return length of entire ACL
    ACLCTX longword unsigned;                          /* ACL context field
    ACLSTS longword unsigned;                          /* ACL return err status

    FILL_10 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_11 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_12 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_13 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_14 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_15 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_16 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_17 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_18 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_19 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_20 longword fill prefix XABPRODEF tag $$;     /* spare
    FILL_21 longword fill prefix XABPRODEF tag $$;     /* spare

    constant PROLEN equals . prefix XAB$ tag K;        /* xabpro length
    constant PROLEN equals . prefix XAB$ tag C;        /* xabpro length

    end XABPRODEF1;

end_module $XABPRODEF;
```

```
module $XABTRMDEF;
/*--
/*++
/*
/*        terminal control xab field definitions
/*                $xabtrmdef
/*
/*
/*
constant TRM    equals 31  prefix XAB tag $C;           /*XABTRM ID CODE

aggregate XABTRMDEF structure prefix XAB$;
    FILL_1 byte fill prefix XABTRMDEF tag $$;
    FILL_2 byte fill prefix XABTRMDEF tag $$;
    FILL_3 word fill prefix XABTRMDEF tag $$;
    FILL_4 longword fill prefix XABTRMDEF tag $$;       /*HAS SAME COD, BLN, SPARE AND NXT FIELD
                                                        /*THESE 4 FIELDS ARE COMMON TO ALL XABS AND
                                                        /*HAVE BEEN DEFINED BY $XABDEF
    ITMLST longword unsigned;                           /* item list address
    ITMLST_LEN word unsigned;                           /* item list length
    FILL_5 word fill prefix XABTRMDEF tag $$;           /* spare
    FILL_6 longword fill prefix XABTRMDEF tag $$;       /* spare
    FILL_7 longword fill prefix XABTRMDEF tag $$;       /* spare
    FILL_8 longword fill prefix XABTRMDEF tag $$;       /* spare
    FILL_9 longword fill prefix XABTRMDEF tag $$;       /* spare
    FILL_10 longword fill prefix XABTRMDEF tag $$;      /* spare
    constant TRMLEN equals . prefix XAB$ tag K;         /* length of xab of type terminal control
    constant TRMLEN equals . prefix XAB$ tag C;         /* length of xab of type terminal control
end XABTRMDEF;

end_module $XABTRMDEF;
```

**F

```
module $XABSUMDEF;
/*--
/*++
/*
/*        summary xab field definitions
/*              $xabsumdef
/*
/*
constant SUM    equals 22  prefix XAB tag $C;          /* xabsum id code

aggregate XABSUMDEF structure prefix XAB$;
    FILL_1 byte fill prefix XABSUMDEF tag $$;
    FILL_2 byte fill prefix XABSUMDEF tag $$;
    FILL_3 word fill prefix XABSUMDEF tag $$;
    FILL_4 longword fill prefix XABSUMDEF tag $$;      /*HAS SAME COD, BLN, SPARE AND NXT FIELD
                                                       /*THESE 4 FIELDS ARE COMMON TO ALL XABS AND
                                                       /*HAVE BEEN DEFINED BY $XABDEF
    NOA byte unsigned;                                 /* number of defined areas for index file
    NOK byte unsigned;                                 /* number of defined keys for index file
    PVN word unsigned;                                 /* prologue version number (relative and index files)
    constant SUMLEN equals . prefix XAB$ tag K;        /* xabsum length
    constant SUMLEN equals . prefix XAB$ tag C;        /* xabsum length
end XABSUMDEF;

end_module $XABSUMDEF;
```

```
module $XABKEYDEF;
/*--
/*++
/*
/*        key definition xab field definitions
/*              $xabkeydef
/*
/*
constant KEY    equals 21  prefix XAB tag $C;           /* xabkey id code

aggregate XABKEYDEF structure prefix XAB$;
    FILL_1 byte fill prefix XABKEYDEF tag $$;
    FILL_2 byte fill prefix XABKEYDEF tag $$;
    FILL_3 word fill prefix XABKEYDEF tag $$;
    FILL_4 longword fill prefix XABKEYDEF tag $$;       /*HAS SAME COD, BLN, SPARE AND NXT FIELD
                                                        /*THESE 4 FIELDS ARE COMMON TO ALL XABS AND
                                                        /*HAVE BEEN DEFINED BY $XABDEF
/*
/* the field layout of the key xab is such that it matchs as
/* closely as possible the layout of a key decriptor in the
/* index file prologue. this is so the contents may be moved
/* between the two structures as efficiently as possible.
/*
    IAN byte unsigned;                                  /* index level area number
    LAN byte unsigned;                                  /* lowest index level area number
    DAN byte unsigned;                                  /* data level area number
    LVL byte unsigned;                                  /* level of root bucket
    IBS byte unsigned;                                  /* size of index buckets in virtual blocks
    DBS byte unsigned;                                  /* size of data buckets in virtual blocks
    RVB longword unsigned;                              /* root bucket start vbn
    FLG_OVERLAY union;
        FLG byte unsigned;                              /* key option flags
        FLG_BITS0 structure;
            DUP bitfield mask;                          /* duplicate key values allowed
            CHG bitfield mask;                          /* alt key only --key field may change on update
            NUL bitfield mask;                          /* alt key only --null key value enable
            IDX_NCMPR bitfield mask;                    /* indicate index records for given key are not compressed
            FILL_5 bitfield length 2 fill prefix XABKEYDEF tag $$;/* spare
            KEY_NCMPR bitfield mask;                    /* indicates key is not compressed in data record
        end FLG_BITS0;

        FLG_BITS1 structure;
            FILL_6 bitfield fill prefix XABKEYDEF tag $$;/* space over dup
            FILL_7 bitfield length 2 fill prefix XABKEYDEF tag $$;/* spare
            FILL_8 bitfield fill prefix XABKEYDEF tag $$;/* space over idx_ncmpr
            FILL_9 bitfield length 2 fill prefix XABKEYDEF tag $$;/* spare
            FILL_10 bitfield fill prefix XABKEYDEF tag $$;/* space over key_ncmpr
            DAT_NCMPR bitfield mask;                    /* data record is not compressed
        end FLG_BITS1;

    end FLG_OVERLAY;
    DTP byte unsigned;                                  /* key field data type
    constant STG        equals 0  prefix XAB tag $C;    /* string
    constant IN2        equals 1  prefix XAB tag $C;    /* signed 15 bit integer (2 bytes)
    constant BN2        equals 2  prefix XAB tag $C;    /* 2 byte binary
    constant IN4        equals 3  prefix XAB tag $C;    /* signed 31 bit integer (4 bytes)
    constant BN4        equals 4  prefix XAB tag $C;    /* 4 byte binary
    constant PAC        equals 5  prefix XAB tag $C;    /* packed decimal (1-16 bytes)
    constant IN8        equals 6  prefix XAB tag $C;    /* signed 63 bit integer (4 bytes)
    constant BN8        equals 7  prefix XAB tag $C;    /* 8 byte binary
    constant MAXDTP     equals 7  prefix XAB tag $C;    /* max. legal data type
    NSG byte unsigned;                                  /* number of key segments
    NUL byte unsigned;                                  /* nul key character
    TKS byte unsigned;                                  /* total key field size (bytes)
    "REF" byte unsigned;                                /* key of reference (0=prim key,
                                                        /* 1-254 = alternate keys)
    MRL word unsigned;                                  /* minimun record length to contain key field
    IFL word unsigned;                                  /* index bucket fill size (bytes)
    DFL word unsigned;                                  /* data bucket fil size (bytes)
    POS_OVERLAY union;
        POS word unsigned dimension 8;                  /* key field record offset positions
        POS_FIELDS structure;
            POS0 word unsigned;                         /* segment 0
            POS1 word unsigned;                         /* segment 1
            POS2 word unsigned;                         /* segment 2
            POS3 word unsigned;                         /* segment 3
            POS4 word unsigned;                         /* segment 4
            POS5 word unsigned;                         /* segment 5
            POS6 word unsigned;                         /* segment 6
            POS7 word unsigned;                         /* segment 7
        end POS_FIELDS;
    end POS_OVERLAY;
    SIZ_OVERLAY union;
        SIZ byte unsigned dimension 8;                  /* key field segment sizes
        SIZ_FIELDS structure;
            SIZ0 byte unsigned;                         /* segment 0
            SIZ1 byte unsigned;                         /* segment 1
            SIZ2 byte unsigned;                         /* segment 2
            SIZ3 byte unsigned;                         /* segment 3
            SIZ4 byte unsigned;                         /* segment 4
            SIZ5 byte unsigned;                         /* segment 5
            SIZ6 byte unsigned;                         /* segment 6
            SIZ7 byte unsigned;                         /* segment 7
        end SIZ_FIELDS;
    end SIZ_OVERLAY;
    FILL_11 word fill prefix XABKEYDEF tag $$;          /* spare
/*
/* the positions of the above fields are dictated by the key descriptor
/* record layout in the index file prologue.
/*
    KNM longword unsigned;                              /* pointer to 32 character key name buffer
    DVB longword unsigned;                              /* first data bucket start vbn
    constant KEYLEN_V2 equals . prefix XAB$ tag K;      /* old xabkey length
    constant KEYLEN_V2 equals . prefix XAB$ tag C;      /* old xabkey length
/*
/* Additions for prologue 3 files
/*
    TYP_OVERLAY union;
        TYP byte unsigned dimension 8;                  /* key field segment types
        TYP_FIELDS structure;
            TYP0 byte uns[illegible]ed;                 /* segment 0
            TYP1 byte u[illegible]ed;                   /* segment 1
            TYP2 byte unsigned;                              /* segment 2
            TYP3 byte unsigned;                              /* segment 3
            TYP4 byte unsigned;                              /* segment 4
            TYP5 byte unsigned;                              /* segment 5
            TYP6 byte unsigned;                              /* segment 6
            TYP7 byte unsigned;                              /* segment 7
        end TYP_FIELDS;
    end TYP_OVERLAY;
    PROLOG byte unsigned;                                    /* indicate prologue version desired (primary key only)
    constant PRG3       equals 3  prefix XAB tag $C;         /* Prologue version three
    constant PRG2       equals 2  prefix XAB tag $C;         /* Prologue version two
    constant PRG1       equals 1  prefix XAB tag $C;         /* Prologue versoin one
    FILL_12 byte fill prefix XABKEYDEF tag $$;               /* spare
    FILL_13 word fill prefix XABKEYDEF tag $$;               /* spare
    constant KEYLEN equals . prefix XAB$ tag K;              /* xabkey length
    constant KEYLEN equals . prefix XAB$ tag C;              /* xabkey length

/*--
/*++
end XABKEYDEF;

end_module $XABKEYDEF;
```

```
module $XABCXFDEF;
/*
/*       RMS Context XAB associated with the FAB
/*               $xabcxfdef
/*
/*

aggregate XABCXFDEF structure prefix XAB$;
    FILL_1 byte fill prefix XABCXFDEF tag $$;           /* COD - xab id code
    constant CXF        equals 32  prefix XAB tag $C;   /* XABCXF id code
    FILL_2 byte fill prefix XABCXFDEF tag $$;           /* BLN - block length
    FILL_3 word fill prefix XABCXFDEF tag $$;           /* (spare)
    FILL_4 longword fill prefix XABCXFDEF tag $$;       /* NXT - xab chain link
                                                        /* UP TILL NOW COMMON AMONG ALL XABS
/*
/*       Following in common with the CXR block, too.
/*       Do not rearrange without changing both.
/*
    CXFSTS longword unsigned;                           /* Status of the last file operation.
    CXFSTV longword unsigned;                           /* Status Value of the last file operation.
/*
/* Top four bits of the options longword are reserved for the XABCXR.  These
/* bits describe the version of the key buffer.
/*
    CXFCOP_OVERLAY union;
        CXFCOP longword unsigned;                       /* Context Options.
        CXFCOP_BITS structure;
            CXFRST bitfield mask;                       /* Restore file state - use context blk as input.
        end CXFCOP_BITS;
    end CXFCOP_OVERLAY;
    CXFBKP longword unsigned;                           /* Bookkeeping bits
    CXFIFI word unsigned;                               /* Internal File Identifier
    CXFVER byte unsigned;                               /* prologue version num
    FILL_5 byte fill prefix XABCXFDEF tag $$;           /* spare to longword align commonality
    FILL_6 longword fill prefix XABCXFDEF tag $$;       /* spare
/*
/*       Up Till now in common with XABCXR, too.
/*

/*       The following fields correspond to those in the FAB or IFB
/*       They should not be rearranged as their order is assumed for
/*       purposes of moving large chunks of data rather than a byte
/*       or word at a time.  Note: ASSUME is used in the actual code
/*
    CXFDEQ word unsigned;                               /* Default extention quantity
    CXFFAC byte unsigned;                               /* File access
    CXFSHR byte unsigned;                               /* File Sharing
    CXFRTE word unsigned;                               /* (Not used)
    FILL_7 byte fill prefix XABCXFDEF tag $$;           /* spare
    CXFORG byte unsigned;                               /* file organization
    CXFGBC word unsigned;                               /* global buffer count
    CXFRTV byte unsigned;                               /* retrieval window
    FILL_8 byte fill prefix XABCXFDEF tag $$;
    FILL_9 longword dimension 4 fill prefix XABCXFDEF tag $$;/* spares
    constant CXFLEN equals . prefix XAB$ tag K;         /* length of xab type CXF
    constant CXFLEN equals . prefix XAB$ tag C;         /* length of xab type CXF
```

```
end XABCXFDEF;
end_module $XABCXFDEF;
```

```
module $XABCXRDEF;
/*
/*          RMS Context XAB associated with the RAB
/*                   $xabcxrdef
/*
/*

aggregate XABCXRDEF structure prefix XAB$;
    FILL_1 byte fill prefix XABCXRDEF tag $$;           /* COD - xab id code
    constant CXR        equals 33  prefix XAB tag $C;   /* XABCXR id code
    FILL_2 byte fill prefix XABCXRDEF tag $$;           /* BLN - block length
    FILL_3 word fill prefix XABCXRDEF tag $$;           /* (spare)
    FILL_4 longword fill prefix XABCXRDEF tag $$;       /* NXT - xab chain link
                                                        /* UP TILL NOW COMMON AMONG ALL XABS
/*
/*          Following in common with the CXF block, too.
/*          Do not rearrange without changing it.
/*
    CXRSTS longword unsigned;                           /* Status of the last record operation.
    CXRSTV longword unsigned;                           /* Status Value of the last record operation.
    CXRCOP_OVERLAY union;
        CXRCOP longword unsigned;                       /* Context Options.
        CXRCOP_BITS structure;
            CXRRST bitfield mask;                       /* Restore file/record state - use context blk as input.
            FILL_5 bitfield length 27 fill prefix XABCXRDEF TAG $$;
            CXRBVER bitfield length 4;                  /* Version of Key buffer
            constant CXB_VER1 equals 1 prefix XAB tag $C;
        end CXRCOP_BITS;
    end CXRCOP_OVERLAY;
    CXRBKP longword unsigned;                           /* Bookkeeping bits
    CXRISI word unsigned;                               /* Internal Record Identifier
    CXRVER byte unsigned;                               /* prologue version num.
    FILL_6 byte fill prefix XABCXRDEF tag $$;           /* spare to longword align commonality
    FILL_7 longword fill prefix XABCXRDEF tag $$;       /* spare
/*
/*          Up Till now in common with XABCXF, too.
/*

/*
/*          The following elements are arranged such that large amounts of
/*          data can be moved at a time rather than words or bytes.  Do not
/*          rearrange them without this consideration in mind.
/*
/*          The following elements are stream dependent regardless of file org.
/*
    CXRMBF byte unsigned;                               /* Multibuffer count
    CXRMBC byte unsigned;                               /* Multiblock count
    CXRBFZ word unsigned;                               /* sz in byte of CXRBUF
/*
/*       The following elements are necessary for saving the NRP context for
/*       Sequential and Relative files.
/*
    CXRVBN longword unsigned;                           /* NRP VBN
    CXROFF word unsigned;                               /* NRP offset in VBN
    FILL_8 word unsigned;                               /* mbz - longword align
/*
```

```
/*      The following elements are necessary for saving the NRP context for
/*      ISAM files.
/*
    CXRPOS0 longword unsigned;                          /* Primary Positioning RFA
    CXRPOS4 word unsigned;
    FILL_9 word fill prefix XABCXRDEF tag $$;           /* Spare MBZ
    CXRCUR0 longword unsigned;                          /* Current Positioning RFA
    CXRCUR4 word unsigned;
    FILL_10 word fill prefix XABCXRDEF tag $$;          /* Spare MBZ
    CXRSID0 longword unsigned;                          /* SIDR positioning RFA
    CXRSID4 word unsigned;
    FILL_11 word fill prefix XABCXRDEF tag $$;          /* Spare MBZ
    CXRCNT word unsigned;                               /* SIDR array count
    CXRKREF byte unsigned;                              /* Cur Key of Reference
    CXRKLEN byte unsigned;                              /* Length of key
    CXRBUF longword unsigned;                           /* address of key buf
    constant CXRBLEN equals 512 prefix XAB tag $C;      /* Length of CXRBUF (bytes)
    FILL_12 longword dimension 2 fill prefix XABCXRDEF tag $$;/* Spares
    constant CXRLEN equals . prefix XAB$ tag K;         /* Length of XAB type CXR
    constant CXRLEN equals . prefix XAB$ tag C;         /* Length of XAB type CXR
end XABCXRDEF;

end_module $XABCXRDEF;
```

```
module $XABJNLDEF;
/*++
/*
/*       Journal XAB definitions
/*               $xabjnldef
/*
/*
constant JNL    equals 34  prefix XAB tag $C;           /* xabjnl id code

aggregate XABJNLDEF structure prefix XAB$;
    FILL_1 byte fill prefix XABJNLDEF tag $$;
    FILL_2 byte fill prefix XABJNLDEF tag $$;
    FILL_3 word fill prefix XABJNLDEF tag $$;
    FILL_4 longword fill prefix XABJNLDEF tag $$;       /*HAS SAME COD, BLN, SPARE AND NXT FIELD
                                                        /*THESE 4 FIELDS ARE COMMON TO ALL XABS AND
                                                        /*HAVE BEEN DEFINED BY $XABDEF

    JOP_OVERLAY union;
        JOP word unsigned;                              /* journaling flags
        JOP_BITS structure;
            ONLY_RU bitfield mask;                      /* Recovery-unit only access
            RU bitfield mask;                           /* Recovery unit
            BI bitfield mask;                           /* Before Image
            AI bitfield mask;                           /* After Image
            AT bitfield mask;                           /* Audit Trail
            NEVER_RU bitfield mask;                     /* Never journal in Recovery-unit
        end JOP_BITS;
    end JOP_OVERLAY;

    FILL_5 word fill prefix XABJNLDEF tag $$;

    BIS byte unsigned;                                  /* BI journal name buffer size
    BIL byte unsigned;                                  /* BI journal name return size
    FILL_6 word fill prefix XABJNLDEF tag $$;
    BIA longword unsigned;                              /* BI journal name buffer address

    AIS byte unsigned;                                  /* AI journal name buffer size
    AIL byte unsigned;                                  /* AI journal name return size
    FILL_7 word fill prefix XABJNLDEF tag $$;
    AIA longword unsigned;                              /* AI journal name buffer address

    ATS byte unsigned;                                  /* AT journal name buffer size
    ATL byte unsigned;                                  /* AT journal name return size
    FILL_8 word fill prefix XABJNLDEF tag $$;
    ATA longword unsigned;                              /* AT journal name buffer address

    FILL_9 longword fill prefix XABJNLDEF tag $$;
    FILL_10 longword fill prefix XABJNLDEF tag $$;
    FILL_11 longword fill prefix XABJNLDEF tag $$;
    FILL_12 longword fill prefix XABJNLDEF tag $$;
    FILL_13 longword fill prefix XABJNLDEF tag $$;
    FILL_14 longword fill prefix XABJNLDEF tag $$;

    constant MAXJNLNAM equals 16 prefix XAB$ tag K;     /* max size of ascii string journal name
    constant MAXJNLNAM equals 16 prefix XAB$ tag C;     /* max size of ascii string journal name
    constant JNLLEN equals . prefix XAB$ tag K;
    constant JNLLEN equals . prefix XAB$ tag C;
end XABJNLDEF;

end_module $XABJNLDEF;
```

```
module $FSCNDEF;
/*++
/*
/*        Descriptor codes for SYS$FILESCAN
/*
/*

aggregate FLDFLAGS structure prefix FSCN$;
        NODE    bitfield mask;                  /* Node name present
        DEVICE  bitfield mask;                  /* Device name present
        ROOT    bitfield mask;                  /* Root directory present
        DIRECTORY bitfield mask;                /* Directory present
        NAME    bitfield mask;                  /* File name present
        TYPE    bitfield mask;                  /* File type present
        VERSION bitfield mask;                  /* File version present
end FLDFLAGS;

aggregate FSCNDEF structure prefix FSCN$;
    LENGTH word unsigned;                               /* return length word
    ITEM_CODE word unsigned;                            /* item code value
    ADDR longword unsigned;                             /* return length pointer

    constant FILESPEC equals    1 prefix FSCN tag $;    /* complete filespec
    constant NODE equals        2 prefix FSCN tag $;    /* node:: field
    constant DEVICE equals      3 prefix FSCN tag $;    /* device: field
    constant ROOT equals        4 prefix FSCN tag $;    /* [root.] field
    constant DIRECTORY equals   5 prefix FSCN tag $;    /* [directory] field
    constant NAME equals        6 prefix FSCN tag $;    /* name field
    constant TYPE equals        7 prefix FSCN tag $;    /* .typ field
    constant VERSION equals     8 prefix FSCN tag $;    /* ;version field

    constant ITEM_LEN equals . prefix FSCN$ tag S;
end FSCNDEF;

end_module $FSCNDEF;
```

```
module $RMEDEF;
/*
/*                  rms escape definitions
/*
/*  the following values identify various requests for non-standard rms
/*  functions.  they are currently input to the $modify function in the
/*  ctx field of the fab only if the esc bit is set in fop.  incorrect
/*  use of these capabilties could cause rms to fail, hence great caution
/*  should be exercised in their use.
/*

constant SETRFM equals 1  prefix RME tag $C;           /* change rfm, mrs, and fsz (if vfc) in ifab only
constant PPFECHO equals 2  prefix RME tag $C;          /* enable echo of SYS$INPUT to SYS$OUTPUT

end_module $RMEDEF;
```

0313 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RMSFILSTR
SDL

RMSMAC
REQ

RMSINTSTR
SDL

RMSUSR
SDL

NWADEF
MDL

RMSFWADEF
SDL

RMSSHR
SDL