H 5

```
   SSSSSSSSSSSS     OOOOOOOOO     RRRRRRRRRRRR     TTTTTTTTTTTTTTT     333333333        222222222
   SSSSSSSSSSSS     OOOOOOOOO     RRRRRRRRRRRR     TTTTTTTTTTTTTTT     333333333        222222222
   SSSSSSSSSSSS     OOOOOOOOO     RRRRRRRRRRRR     TTTTTTTTTTTTTTT     333333333        222222222
SSS              OOO         OOO  RRR         RRR        TTT        333         333  222         222
SSS              OOO         OOO  RRR         RRR        TTT        333         333  222         222
SSS              OOO         OOO  RRR         RRR        TTT        333         333  222         222
SSS              OOO         OOO  RRR         RRR        TTT                    333              222
SSS              OOO         OOO  RRR         RRR        TTT                    333              222
SSS              OOO         OOO  RRR         RRR        TTT                    333              222
   SSSSSSSSS     OOO         OOO  RRRRRRRRRRRR           TTT                 333              222
   SSSSSSSSS     OOO         OOO  RRRRRRRRRRRR           TTT                 333              222
   SSSSSSSSS     OOO         OOO  RRRRRRRRRRRR           TTT                 333              222
            SSS  OOO         OOO  RRR   RRR              TTT                    333        222
            SSS  OOO         OOO  RRR   RRR              TTT                    333        222
            SSS  OOO         OOO  RRR   RRR              TTT                    333        222
            SSS  OOO         OOO  RRR      RRR           TTT        333         333     222
            SSS  OOO         OOO  RRR      RRR           TTT        333         333     222
            SSS  OOO         OOO  RRR      RRR           TTT        333         333     222
SSSSSSSSSSSS        OOOOOOOOO     RRR         RRR        TTT           333333333     222222222222222
SSSSSSSSSSSS        OOOOOOOOO     RRR         RRR        TTT           333333333     222222222222222
SSSSSSSSSSSS        OOOOOOOOO     RRR         RRR        TTT           333333333     222222222222222
```

_$2

Pse

SOR

SOR

SOR

SOR

_LI

```
**FILE**ID**SRTSPC
```

```
  SSSSSSSS  RRRRRRRR    TTTTTTTTTT    SSSSSSSS  PPPPPPPP      CCCCCCCC
  SSSSSSSS  RRRRRRRR    TTTTTTTTTT    SSSSSSSS  PPPPPPPP      CCCCCCCC
SS          RR      RR      TT      SS          PP      PP  CC
SS          RR      RR      TT      SS          PP      PP  CC
SS          RR      RR      TT      SS          PP      PP  CC
SS          RR      RR      TT      SS          PP      PP  CC
  SSSSSS    RRRRRRRR        TT        SSSSSS    PPPPPPPP    CC
  SSSSSS    RRRRRRRR        TT        SSSSSS    PPPPPPPP    CC
        SS  RR  RR          TT              SS  PP          CC
        SS  RR  RR          TT              SS  PP          CC
        SS  RR    RR        TT              SS  PP          CC              ....
        SS  RR    RR        TT              SS  PP          CC              ....
SSSSSSSS    RR      RR      TT      SSSSSSSS    PP            CCCCCCCC      ....
SSSSSSSS    RR      RR      TT      SSSSSSSS    PP            CCCCCCCC      ....

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

```
! File: SRTSPC.REQ IDENT = 'V04-000'    ! File: SRTSPC.REQ Edit: PDG3028
!
!
!****************************************************************************
!*                                                                          *
!*  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-11 SORT/MERGE,  PDP-11 SORT/MERGE
!
! ABSTRACT:
!
!       This require file is for data structures returned from specification
!       file processing.
!       This file is used as a library source.
!
! ENVIRONMENT:  VAX/VMS user mode
!
! AUTHOR: V. Bennison, CREATION DATE: 03-May-1982
!
! MODIFIED BY:
!
!       31-Aug-1982     PDG
!               Add definitions that are required for SRTSPC.BLI.
!       T03-016 Rework TDT table to give precedence to AND/OR.  PDG 13-Dec-1982
!       T03-017 Add WF_NAMES, CFT indices of work file names.  PDG 26-Dec-1982
!       T03-018 Removed RDT_VAR.  PDG 3-Jan-1983
!       T03-019 Removed PT/ST_ADR; added WRK_SIZ, BS_DECM.  PDG 26-Jan-1983
!       T03-020 Add FDT_SCA'E and CA_PAD.  PDG 8-Feb-1983
!       T03-022 Fix computation for packed in KFT_UNITS_.  PDG 11-Feb-1983
!       T03-022 Remove unreferenced fields.  PDG 16-Mar-1983
!       T03-024 Work around Bliss bug with CA_LINKAGE_LB.  PDG 12-May-1983
!       T03-025 Define KFT_NDE_SIZ_ for BLISS16.  PDG 26-Jul-1983
!       T03-026 Put WHILE_FAIL_ here.  PDG 1-Aug-1983
!       T03-027 Word-align elements in CON_SYM_TAB.  PDG 1-Aug-1983
!       T03-028 Make sharing of code easier to maintain.  PDG 31-Jan-1984
```

```
;   0058  0      !--
```

```
:       0     !-----------------------------------------------------------------
:       0     !                 FIELD DEFINITION TABLE (FDT)
:       0     !-----------------------------------------------------------------
:       0     LITERAL
:       0         FDT_MAX = 255,              !maximum number of entries in FDT
:       0         FDT_UNIT = 6;               ! Size in bytes, must be even for bliss16
:       0                                     ! because of 16 bit field
:       0     STRUCTURE
:       0         FDT_TAB[ O,B,P,S,E; BS ] =
:       0             [ BS*FDT_UNIT ]
:       0             ( FDT_TAB + O*FDT_UNIT + B )<P,S,E>;
:       0     MACRO
:       0         FDT_TYPE    = 0, 0,  8, 0 %,                ! Data type
:       0         FDT_SCALE   = 1, 0,  8, 1 %,                ! Scale factor
:       0         FDT_FLD_POS = 2, 0, 16, 0 %,                ! Position of field
:       0         FDT_FLD_SIZ = 4, 0, 16, 0 %;                ! Size of field
:  0      !---------------------------------------------------------------
:  0      !                TEST DEFINITION TABLE (TDT)
:  0      !---------------------------------------------------------------
:  0      LITERAL
:  0          TDT_MAX = 255,              !maximum number of entries in TDT
:  0          TDT_UNIT = 4;                                ! Size in bytes
:  0
:  0      STRUCTURE
:  0          TDT_TAB[ O,B,P,S,E; BS ] =
:  0              [ BS*TDT_UNIT ]
:  0              ( TDT_TAB + O*TDT_UNIT + B )<P,S,E>;
:  0
:  0      MACRO
:  0          TDT_TRUE    =  0, 0, 1, 0 %,        ! Set to simply return TRUE
:  0          TDT_CMP     =  0, 1, 3, 0 %,        ! The comparison flags
:  0          TDT_EQL     =  0, 1, 1, 0 %,        ! True if "Equal to" succeeds
:  0          TDT_LSS     =  0, 2, 1, 0 %,        ! True if "Less than" succeeds
:  0          TDT_GTR     =  0, 3, 1, 0 %,        ! True if "Greater than" succeeds
:  0          TDT_CONSTANT=  0, 4, 1, 0 %,        ! True iff FLD_TWO is to CFT
:  0          TDT_FLD_ONE =  1, 0, 8, 0 %,        ! Index in FDT of 1st field
:  0          TDT_FLD_TWO =  2, 0, 8, 0 %,        ! Index in FDT (or CFT) of 2nd field
:  0          TDT_GOTO    =  3, 0, 8, 0 %;        ! TDT index adjustment
:  0
:  0      ! This structure should only be referenced by the routines that builds it,
:  0      ! and the routine SOR$$TDT!
:  0
:  0      ! This table is used as follows:
:  0      !
:  0      !       Set IX to the index of the test description to test
:  0      !Loop:  If TDT_TRUE is clear then return TRUE
:  0      !       If the comparison between FLD_ONE and FLD_TWO is true
:  0      !           (according to the EQL/LSS7GTR bits)
:  0      !       then
:  0      !           if TDT_GOTO is zero then return false else add TDT_GOTO to IX
:  0      !       else
:  0      !           add 1 to IX
:  0      !       goto Loop
:  0      !
```

```
!-----------------------------------------------------------------
!                   KEY/DATA FIELD TABLE (KFT)
!-----------------------------------------------------------------
LITERAL
    KFT_MAX = 255,              !maximum number of entries in KFT
    KFT_UNIT = 8;               ! Size in bytes, must be even for bliss16
                                ! because of 16 bit field

STRUCTURE
    KFT_TAB[ O,B,P,S,E; BS ] =
        [ BS*KFT_UNIT ]
        ( KFT_TAB + O*KFT_UNIT + B )<P,S,E>;

MACRO
    KFT_NDE_POS =  0, 0, 16, 0 %,              ! Starting position in node

    KFT_CONTINUE =  3, 0, 1, 0 %,              ! Continue = 1
    KFT_CONSTANT =  3, 1, 1, 0 %,              ! True iff FDT_IDX is to CFT
    KFT_CONT_CDX =  3, 2, 1, 0 %,              ! Continued condition = 1
    KFT_CONDX    =  3, 3, 1, 0 %,              ! Conditional field = 1
    KFT_BUILD    =  3, 4, 1, 0 %,              ! Build the key = 1
    KFT_DESCEND =  3, 5, 1, 0 %,               ! Asc/desc, descend = 1
    KFT_DATA     =  3, 6, 1, 0 %,              ! Key or data, data = 1

    KFT_FDT_IDX =  4, 0,  8, 0 %,              ! Index in FDT (or CFT)
    KFT_TDT_IDX =  5, 0,  8, 0 %,              ! TDT index for forces

    KFT_NDE_SIZ =  6, 0, 16, 0 %;              ! Size (bytes) in internal node
```

```
0      !-----------------------------------------------------------------
0      !                   RECORD DEFINITION TABLE (RDT)
0      !-----------------------------------------------------------------
0      LITERAL
0          RDT_MAX = 64,               !maximum number of entries in RDT
0          RDT_UNIT = 6;               ! Size in bytes
0
0      STRUCTURE
0          RDT_TAB[ O,B,P,S,E; BS ] =
0              [ BS*RDT_UNIT ]
0              ( RDT_TAB + O*RDT_UNIT + B )<P,S,E>;
0
0      MACRO
0          RDT_INCLUDE =  0, 0,  1, 0 %,              ! Include/omit, Include = 1
0          RDT_CONDX   =  0, 1,  1, 0 %,              ! Conditional = 1
0
0          RDT_TDT_IDX =  1, 0,  8, 0 %,              ! Index into TDT
0          RDT_KCT_ADR =  2, 0, 16, 0 %,              ! For Sort-11 only
0          RDT_KFT_IDX =  4, 0,  8, 0 %;              ! Index into KFT
0
0      ! The RDT table is scanned sequentially until either an unconditional entry is
0      ! found, or until a condition (via RDT_TDT_IDX) passes.  This matched entry
0      ! describes whether to omit or include the record (RDT_INCLUDE).  If included,
0      ! then RDT_KFT_IDX is used to index the KFT table, for record reformatting.
0      !
```

```
!--------------------------------------------------------------
!               CONSTANT FIELD TABLE (CFT)
!--------------------------------------------------------------
LITERAL
    CFT_MAX = 255,              !maximum number of entries in CFT
    CFT_UNIT = 2+%BPADDR/8;     ! Size in bytes

STRUCTURE
    CFT_TAB[ O,B,P,S,E; BS ] =
        [ BS*CFT_UNIT ]
        ( CFT_TAB + O*CFT_UNIT + B )<P,S,E>;

MACRO
    CFT_CON_LEN =  0, 0,  8, 0 %,               ! Length of constant
    CFT_CON_ADR =  2, 0, %BPADDR, 0 %;  ! Address of constant
```

```
!---------------------------------------------------------------
!               COMMON DEFINITIONS
!---------------------------------------------------------------

%IF %BLISS(BLISS32)
%THEN
        LIBRARY 'SYS$LIBRARY:STARLET';
        LIBRARY 'SRC$:SORLIB';
%ELSE
        LIBRARY 'S11V3SRC:SMCOM';
%FI


! Define the linkage to the common routines
!
LITERAL
    LB_REG = 4;
LINKAGE
    CA_LINKAGE =
        %BLISS32( CALL:GLOBAL(CA=COM_REG_CTX) ) ! MUST BE SAME AS CAL_CTXREG!
        %BLISS16( JSR ),
    CA_LINKAGE_LB =             ! Same as CA_LINKAGE, with an extra register
        %BLISS32( CALL:GLOBAL(CA=COM_REG_CTX, LB=LB_REG) )
        %BLISS16( JSR :GLOBAL(                LB=LB_REG) ),
    CA_LINK_SEGMENT =
        %BLISS16( JSR )
        %BLISS32( JSB (
                REGISTER=6,
                REGISTER=COM_REG_SRC2):
                GLOBAL(CA=COM_REG_CTX)
                PRESERVE(COM_REG_SRC2)
                NOTUSED(7,8,9)
                NOPRESERVE(0,1,2,3,4,5));


! A macro to declare/get the address of the common area
!
MACRO
    CA_AREA_( X ) =
        %IF %BLISS(BLISS32)
        %THEN
            EXTERNAL REGISTER
                %IF %NULL(X) %THEN CA %ELSE X %FI
                    = COM_REG_CTX:      REF BLOCK[CTX_K_SIZE]
                                        FIELD(CTX_FIE[DS);
        %ELSE
            %IF NOT %NULL(X)
            %THEN
                LOCAL
                    X : REF BLOCK [, %UPVAL] FIELD (COM_FIELDS);
                %QUOTE GET_IMPAREA_( X );
            %FI
        %FI %;

! Specification file error messages
!
!LITERAL
```

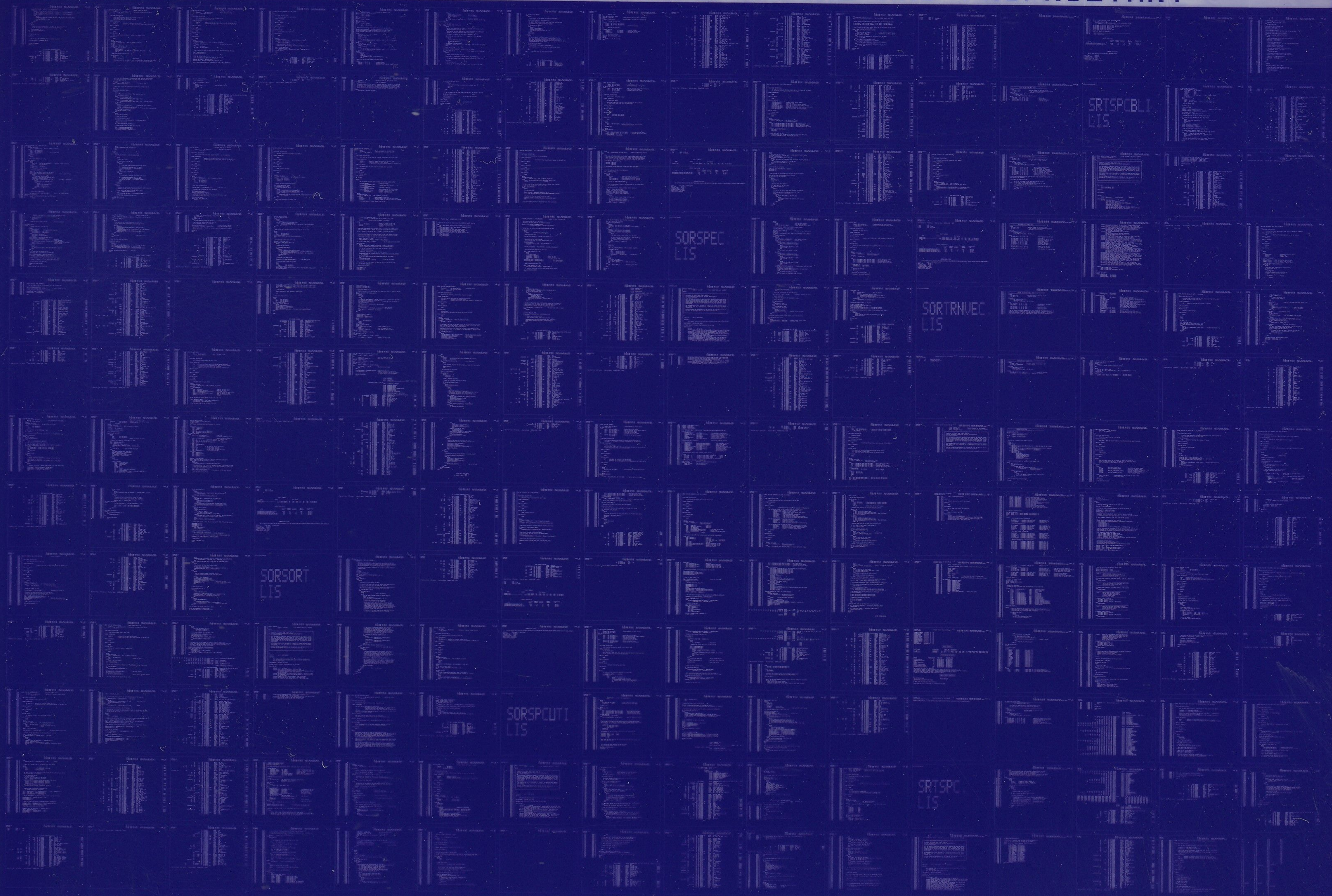
```
;     0     !     SRTIWA = SOR$_SRTIWA,        !insufficient work area
;     0     !     SPCOVR = SOR$_SPCOVR,        !warning: overriden specification
;     0     !     SPCMIS = SOR$_SPCMIS,        !warning: invalid merge specification
;     0     !     SPCSIS = SOR$_SPCSIS,        !warning: invalid sort specification
;     0     !     SPCIVP = SOR$_SPCIVP,        !invalid sort process
;     0     !     SPCIVS = SOR$_SPCIVS,        !invalid specification
;     0     !     SPCIVC = SOR$_SPCIVC,        !invalid collating sequence specification
;     0     !     SPCIVF = SOR$_SPCIVF,        !invalid field specification
;     0     !     SPCIVD = SOR$_SPCIVD,        !invalid data type
;     0     !     SPCIVX = SOR$_SPCIVX,        !invalid condition specification
;     0     !     SPCIVK = SOR$_SPCIVK,        !invalid key or data specification
;     0     !     SPCIVI = SOR$_SPCIVI;        !invalid include or omit specification
;     0
;     0
;     0
;     0     ! A macro to expand fields
;     0     !
; L   0     %IF %BLISS(BLISS32)
;     0     %THEN   MACRO _(X,Y) = %QUOTE %EXPAND %FIELDEXPAND(X) %;
; U   0     %ELSE   MACRO _(X,Y) = %QUOTE %EXPAND %FIELDEXPAND(Y) %;
;     0     %FI
;     0
;     0     MACRO
;     0             ! Sort/Merge process information
;     0             !
;     0             CA_PROCESS      = %EXPAND _(COM_SORT_TYPE,      COM_PROCESS_) %,
;     0             CA_PROCESS_OVR  = %EXPAND _(COM_OVR_PROC,       COM_PROC_OVR_) %,
;     0             CA_VAR_MERGE    = %EXPAND _(COM_MERGE,          COM_MERGE_) %,
;     0
;     0             ! Collating information
;     0             !
;     0             CA_TIE_BREAK    = %EXPAND _(COM_TIE_BREAK,      COM_TIE_BREAK_) %,
;     0             CA_ST_ADR       = %EXPAND _(COM_COLLATE,        COM_CS_TAB_ADR_) %,
;     0             CA_ST_SIZ       = %EXPAND _(COM_ST_SIZ,         COM_CS_TAB_SIZ_) %,
;     0             CA_BS_DECM      = %EXPAND _(COM_BS_DECM,        COM_BS_DECM_) %,
;     0             CA_PAD          = %EXPAND _(COM_PAD,            COM_PAD_CHAR_) %,
;     0
;     0             ! Keys and stable information
;     0             !
;     0             CA_KEY_OVR      = %EXPAND _(COM_OVR_KEY,        COM_KEY_OVR_) %,
;     0             CA_CHKSEQ       = %EXPAND _(COM_SEQ_CHECK,      COM_CH_SEQ_) %,
;     0             CA_CHKSEQ_OVR   = %EXPAND _(COM_SEQ_CHECK,      COM_CHSEQ_OVR_) %,
;     0             CA_STABLE       = %EXPAND _(COM_STABLE,         COM_STABLE_) %,
;     0             CA_STABLE_OVR   = %EXPAND _(COM_STABLE,         COM_STBL_OVR_) %,
;     0             CA_COLSEQ_OVR   = %EXPAND _(COM_OVR_COLSEQ,     COM_CSEQ_OVR_) %,
;     0
;     0             ! Record reformatting, and other tables
;     0             !
;     0             CA_RDT_ADR      = %EXPAND _(COM_RDT_ADR,        COM_RDT_ADR_) %,
;     0             CA_RDT_SIZ      = %EXPAND _(COM_RDT_SIZ,        COM_RDT_SIZ_) %,
;     0             CA_KFT_ADR      = %EXPAND _(COM_KFT_ADR,        COM_KFT_ADR_) %,
;     0             CA_KFT_SIZ      = %EXPAND _(COM_KFT_SIZ,        COM_KFT_SIZ_) %,
;     0             CA_CFT_ADR      = %EXPAND _(COM_CFT_ADR,        COM_CFT_ADR_) %,
;     0             CA_CFT_SIZ      = %EXPAND _(COM_CFT_SIZ,        COM_CFT_SIZ_) %,
;     0             CA_FDT_ADR      = %EXPAND _(COM_FDT_ADR,        COM_FDT_ADR_) %,
;     0             CA_FDT_SIZ      = %EXPAND _(COM_FDT_SIZ,        COM_FDT_SIZ_) %,
;     0             CA_TDT_ADR      = %EXPAND _(COM_TDT_ADR,        COM_TDT_ADR_) %,
;     0295  0          CA_TDT_SIZ      = %EXPAND _(COM_TDT_SIZ,        COM_TDT_SIZ_) %,
;     0296  0
; U 0297  0      %IF %BLISS(BLISS16) %THEN
; U 0298  0          CA_STAT_ADR     = %EXPAND _(0,                  COM_STAT_ADR_) %,       ! user error buffer (address)
; U 0299  0          CA_USR_WRN      = %EXPAND _(0,                  COM_USR_WRN_) %,        ! address of user-written warning ro
; U 0300  0          CA_1ST_SPC_ERR  = %EXPAND _(0,                  COM_1ST_SPC_ERR_) %,    ! first spec fatal error code
; U 0301  0          CA_1ST_SPC_LIN  = %EXPAND _(0,                  COM_1ST_SPC_LIN_) %,    ! first spec error line number
;     0302  0      %FI
;     0303  0          CA_CONST_AREA   = %EXPAND _(COM_CONST_AREA,     COM_CONST_AREA_) %,     ! constant area (address)
;     0304  0          CA_WRK_ADR      = %EXPAND _(COM_WRK_ADR,        COM_WRK_ADR_) %,        ! address of work area
;     0305  0          CA_WRK_END      = %EXPAND _(COM_WRK_END,        COM_WRK_END_) %,        ! address past end of work area
;     0306  0          CA_WF_NAMES     = %EXPAND _(COM_WF_NAMES,       COM_WF_NAMES_) %;       ! counted list of indices into CFT o
;     0307  0
;     0308  0      UNDECLARE %QUOTE _;
;     0309  0
;     0310  0      ! A macro to expand fields
;     0311  0      !
; L 0312  0      %IF %BLISS(BLISS32)
;     0313  0      %THEN   MACRO _(X,Y) = X %;
; U 0314  0      %ELSE   MACRO _(X,Y) = Y %;
;     0315  0      %FI
;     0316  0
;     0317  0      ! Values for datatypes
;     0318  0      ! A negative value indicates that the datatype is not supported
;     0319  0      !
;     0320  0      LITERAL
;     0321  0          DT_T    = _(DSC$K_DTYPE_T,      C$$),   ! Character (text)
;     0322  0          DT_AF   = _(-1,                 A$$),   ! Ascii Floating
;     0323  0          DT_AZ   = _(-1,                 Z$$),   ! Ascii Zoned
;     0324  0          DT_DB   = _(-1,                 L$$),   ! Dibol
;     0325  0          DT_F    = _(DSC$K_DTYPE_F,      F$$),   ! F-floating
;     0326  0          DT_D    = _(DSC$K_DTYPE_D,      F$$),   ! D-floating
;     0327  0          DT_G    = _(DSC$K_DTYPE_G,      -1),    ! G-floating
;     0328  0          DT_H    = _(DSC$K_DTYPE_H,      -1),    ! H-floating
;     0329  0          DT_P    = _(DSC$K_DTYPE_P,      P$$),   ! Packed decimal
;     0330  0          DT_B    = _(DSC$K_DTYPE_B,      B$$),   ! Signed binary
;     0331  0          DT_U    = _(DSC$K_DTYPE_BU,     U$$),   ! Unsigned binary
;     0332  0          DT_NU   = _(DSC$K_DTYPE_NU,     D$$),   ! Decimal unsigned
;     0333  0          DT_NL   = _(DSC$K_DTYPE_NL,     I$$),   ! Decimal leading separate
;     0334  0          DT_NLO  = _(DSC$K_DTYPE_NLO,    K$$),   ! Decimal leading overpunch
;     0335  0          DT_NR   = _(DSC$K_DTYPE_NR,     J$$),   ! Decimal trailing separate
;     0336  0          DT_NRO  = _(DSC$K_DTYPE_NRO,    D$$),   ! Decimal trailing overpunch
;     0337  0          DT_NZ   = _(DSC$K_DTYPE_NZ,     -1);    ! Zoned decimal
;     0338  0
;     0339  0      UNDECLARE %QUOTE _;
;     0340  0
;     0341  0      MACRO
;     0342  0          ! Macro to determine the length in bytes, given a KFT pointer
;     0343  0          ! Note that this is not needed after the spec file parser is called,
;     0344  0          ! since KFT_NDE_SIZ_ gives the same information.
;     0345  0          !
; M 0346  0          KFT_UNITS_(KFT_PTR) =
; M 0347  0              BEGIN
; M 0348  0              LOCAL
; M 0349  0                  FDT_IX;
; M 0350  0              FDT_IX = .KFT_PTR[0,KFT_FDT_IDX];
; M 0351  0              IF .KFT_PTR[0,KFT_CONSTANT]
; M 0352  0                        THEN
; M 0353  0                            .CFT[.FDT_IX, CFT_CON_LEN]
; M 0354  0                        ELSE
; M 0355  0                            %IF %BLISS(BLISS32) %THEN
; M 0356  0                            IF .FDT[.FDT_IX, FDT_TYPE] EQL DT_P
; M 0357  0                            THEN
; M 0358  0                                .FDT[.FDT_IX, FDT_FLD_SIZ]/2 + 1     ! Length in bytes
; M 0359  0                            ELSE
; M 0360  0                            %FI
; M 0361  0                                .FDT[.FDT_IX, FDT_FLD_SIZ]
;   0362  0                        END %;
;   0363  0
; L 0364  0            %IF %BLISS(BLISS32)
;   0365  0            %THEN
;   0366  0
;   0367  0            ! Character codes
;   0368  0            !
;   0369  0            LITERAL
;   0370  0                C_LBRACK    = %X'5B',       ! Character "["
;   0371  0                C_RBRACK    = %X'5D',       ! Character "]"
;   0372  0                C_SLASH     = %X'2F',       ! Character "/"
;   0373  0                C_EXCLAM    = %X'21',       ! Character "!"
;   0374  0                C_PERCENT   = %X'25',       ! Character "%"
;   0375  0                C_COMMA     = %X'2C',       ! Character ","
;   0376  0                C_NULL      = %X'00',       ! Character ''
;   0377  0                C_QUOTE     = %X'22',       ! Character '""""'
;   0378  0                C_L_PAREN   = %X'28',       ! Character "("
;   0379  0                C_R_PAREN   = %X'29',       ! Character ")"
;   0380  0                C_COLON     = %X'3A',       ! Character ":"
;   0381  0                C_EQUAL     = %X'3D',       ! Character "="
;   0382  0                C_LESS      = %X'3C',       ! Character "<"
;   0383  0                C_GREATER   = %X'3E',       ! Character ">"
;   0384  0                C_DASH      = %X'2D',       ! Character "-"
;   0385  0                C_SPACE     = %X'20',       ! Character " "
;   0386  0                C_TAB       = %X'09',       ! Character HT
;   0387  0                C_CR        = %X'0D',       ! Character CR
;   0388  0                C_LF        = %X'0A';       ! Character LF
;   0389  0            %FI
;   0390  0            LITERAL
;   0391  0                C_OCT       = %X'6F',       ! Lower case "o", for octal number base
;   0392  0                C_DEC       = %X'64',       ! Lower case "d", for decimal number base
;   0393  0                C_HEX       = %X'78';       ! Lower case "x", for hexadecimal number base
; U   0     %IF %BLISS(BLISS16) %THEN
; U   0     !-------------------------------------------------------------
; U   0     !                 KEY COMPARISON TABLE (KCT)
; U   0     !
; U   0     !       This table is used by Sort-11 for fast access to the
; U   0     !       key descriptions of keys that need to be compared.
; U   0     !
; U   0     !-------------------------------------------------------------
; U   0     LITERAL
; U   0         KCT_MAX = 64,                 !maximum number of entries in KCT
; U   0         KCT_UNIT = 8;                 !size in bytes
; U   0
; U   0     STRUCTURE
; U   0         KCT_TAB[ O,B,P,S,E; BS ] =
; U   0             [ BS*KCT_UNIT ]
; U   0             ( KCT_TAB + O*KCT_UNIT + B )<P,S,E>;
; U   0
; U   0     MACRO
; U   0         KCT_CMP_ADR_  =  0, 0, 16, 0 %,            !address of comparison routine
; U   0         KCT_KEY_POS_  =  2, 0, 16, 0 %,            !starting position of key field
; U   0         KCT_KEY_LEN_  =  4, 0, 16, 0 %,            !length of key field
; U   0         KCT_CONTINUE_ =  6, 0,  1, 0 %,            !continue word
; U   0         KCT_DESCEND_  =  6, 1,  1, 0 %,            !descend = 1, ascend = 0
; U   0         KCT_TYPE_     =  7, 0,  8, 0 %;            !data type, used to reinitialize
;     0     %FI
```

```
! WHILE_FAIL
!       This macro produces code that advances a table pointer through
!       successive entries until the entry is unconditional, or the
!       entry is conditional and passes the condition.
!       The parameter to this macro (X) is the identification of the table.
!       The table pointer must be of the form (X)_PTR, and the table must
!       have the following fields: (X)_CONDX and (X)_TDT_IDX.
!
MACRO
    WHILE_FAIL_(X) =
        BEGIN
        MACRO
            X_PTR       = %NAME(X,'_PTR') %QUOTE %,
            X_CONDX     = %NAME(X,'_CONDX') %QUOTE %,
            X_TDT_IDX   = %NAME(X,'_TDT_IDX') %QUOTE %;
        !
        ! While we fail conditional tests
        !
        WHILE 1 DO
            BEGIN
            LOCAL
                PASS;

            ! Unconditional tests are easy
            !
            IF NOT .X_PTR[0, X_CONDX] THEN EXITLOOP;

            ! We have a condition
            !
            PASS = %IF %BLISS(BLISS32) %THEN SOR$$TDT %ELSE $TDT %FI (
                INPREC[0],                      ! Length/address of record
                TDT[.X_PTR[0,X_TDT_IDX],BASE_]  ! Address of TDT tests
                );
            IF .PASS GTRU 1 THEN RETURN .PASS;  ! Unexpected result
            IF .PASS EQLU 1 THEN EXITLOOP;      ! We passed the test!

            ! Advance to the next record definition
            !
            X_PTR = X_PTR[1,BASE_];
            END;
        END %;
```

```
; U 0460  0      %IF %BLISS(BLISS16) %THEN
; U 0461  0      !
; U 0462  0      ! Other Sort-11 modules that use the fields defined herein
; U 0463  0      ! like to see underscores at the ends of the names.
; U 0464  0      !
; U 0465  0      MACRO    _(X) = X = %quote %expand %REMAINING %QUOTE % %;
; U 0466  0      MACRO
; U 0467  0              _(FDT_TYPE_,     FDT_TYPE),
; U 0468  0              _(FDT_FLD_POS_, FDT_FLD_POS),
; U 0469  0              _(FDT_FLD_SIZ_, FDT_FLD_SIZ),
; U 0470  0              _(KFT_NDE_POS_, KFT_NDE_POS),
; U 0471  0              _(KFT_NDE_SIZ_, KFT_NDE_SIZ),
; U 0472  0              _(KFT_CONTINUE_,KFT_CONTINUE),
; U 0473  0              _(KFT_CONSTANT_,KFT_CONSTANT),
; U 0474  0              _(KFT_CONT_CDX_,KFT_CONT_CDX),
; U 0475  0              _(KFT_CONDX_,    KFT_CONDX),
; U 0476  0              _(KFT_BUILD_,    KFT_BUILD),
; U 0477  0              _(KFT_DESCEND_, KFT_DESCEND),
; U 0478  0              _(KFT_DATA_,     KFT_DATA),
; U 0479  0              _(KFT_FDT_IDX_, KFT_FDT_IDX),
; U 0480  0              _(KFT_TDT_IDX_, KFT_TDT_IDX),
; U 0481  0              _(RDT_INCLUDE_, RDT_INCLUDE),
; U 0482  0              _(RDT_CONDX_,    RDT_CONDX),
; U 0483  0              _(RDT_TDT_IDX_, RDT_TDT_IDX),
; U 0484  0              _(RDT_KCT_ADR_, RDT_KCT_ADR),
; U 0485  0              _(RDT_KFT_IDX_, RDT_KFT_IDX);
; U 0486  0      UNDECLARE %QUOTE _;
;   0487  0      %FI
;    0488  0      !
;    0489  0      !       Check that the fields are large enough
;    0490  0      MACRO
;    0491  0          S_[O,P,S,E] = 1^S %,
; M  0492  0          M_(V,O,P,S,E)[] = %IF V GTRU MINU(1^S-1,S_(%REMAINING)) %THEN
;    0493  0              %WARN(V,' is too large') %FI %;
;    0494  0
;    0495  0      M_(FDT_MAX, CA_FDT_SIZ, TDT_FLD_ONE, TDT_FLD_TWO, KFT_FDT_IDX)
;    0496  0      M_(TDT_MAX, CA_TDT_SIZ, KFT_TDT_IDX, RDT_TDT_IDX)
;    0497  0      M_(KFT_MAX, CA_KFT_SIZ, RDT_KFT_IDX)
;    0498  0      M_(RDT_MAX, CA_RDT_SIZ, 0,0,8,0)
;    0499  0      M_(CFT_MAX, CA_CFT_SIZ, KFT_FDT_IDX)
;    0500  0
;    0501  0      UNDECLARE %QUOTE S_, %QUOTE M_;
;    0502  0
;    0503  0      !-------------------------------------------------------------
;    0504  0      ! End of SRTSPC.REQ



;                             Library Statistics
;
;                                               -------- Symbols --------       Pages       Processing
;       File                                    Total     Loaded   Percent      Mapped      Time
;
;   _$255$DUA28:[SYSLIB]STARLET.L32;1            9776        14         0         581         00:01.0
;   _$255$DUA28:[SORT32.SRC]SORLIB.L32;1          409       100        24          34         00:00.4




;                                      COMMAND QUALIFIERS

;       BLISS SRC$:SRTSPC/LIS=LIS$:SRTSPC/LIB=SRC$:SRTSPC

; Run Time:          00:06.5
; Elapsed Time:      00:26.3
; Lines/CPU Min:      4623
; Lexemes/CPU-Min: 30036
; Memory Used:  71 pages
; Library Precompilation Complete
```

0366 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



SRTSPCBLI
LIS

SORSPEC
LIS

SORTRNVEC
LIS

SORSORT
LIS

SORSPCUTI
LIS

SRTSPC
LIS