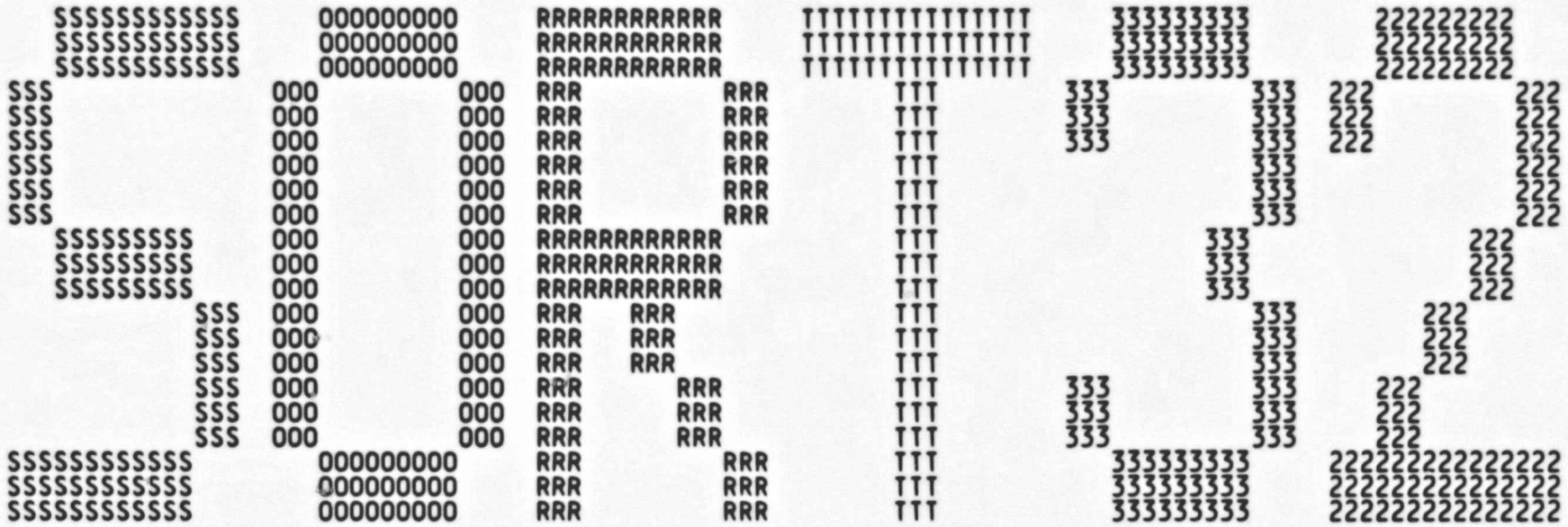
SSSSSSSSSSS 000000000 RRRRRRRRRRR TTTTTTTTTTTTTTT 333333333 222222222
SSSSSSSSSSS 000000000 RRRRRRRRRRR TTTTTTTTTTTTTTT 333333333 222222222
SSSSSSSSSSS 000000000 RRRRRRRRRRR TTTTTTTTTTTTTTT 333333333 222222222
SSS 000 000 RRR RRR TTT 333 333 222 222
SSS 000 000 RRR RRR TTT 333 333 222 222
SSS 000 000 RRR RRR TTT 333 222 222
SSS 000 000 RRR RRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 222
SSSSSSSSS 000 000 RRRRRRRRRRR TTT 333 222
SSSSSSSSS 000 000 RRRRRRRRRRR TTT 333 222
SSSSSSSSS 000 000 RRRRRRRRRRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 333 222
SSS 000 000 RRR RRR TTT 333 333 222
SSS 000 000 RRR RRR TTT 333 333 222
SSSSSSSSSSS 000000000 RRR RRR TTT 333333333 222222222222222
SSSSSSSSSSS 000000000 RRR RRR TTT 333333333 222222222222222
SSSSSSSSSSS 000000000 RRR RRR TTT 333333333 222222222222222

```
  SSSSSSSS    OOOOOO    RRRRRRRR     IIIIII    NN      NN  TTTTTTTTTT  EEEEEEEEEE  RRRRRRRR    FFFFFFFFFF
  SSSSSSSS    OOOOOO    RRRRRRRR     IIIIII    NN      NN  TTTTTTTTTT  EEEEEEEEEE  RRRRRRRR    FFFFFFFFFF
SS           OO      OO  RR      RR     II     NN      NN      TT      EE          RR      RR  FF
SS           OO      OO  RR      RR     II     NN      NN      TT      EE          RR      RR  FF
SS           OO      OO  RR      RR     II     NNNN    NN      TT      EE          RR      RR  FF
SS           OO      OO  RR      RR     II     NNNN    NN      TT      EE          RR      RR  FF
  SSSSSS     OO      OO  RRRRRRRR       II     NN  NN  NN      TT      EEEEEEEE    RRRRRRRR    FFFFFFFF
  SSSSSS     OO      OO  RRRRRRRR       II     NN  NN  NN      TT      EEEEEEEE    RRRRRRRR    FFFFFFFF
        SS   OO      OO  RR   RR        II     NN    NNNN      TT      EE          RR   RR     FF
        SS   OO      OO  RR   RR        II     NN    NNNN      TT      EE          RR   RR     FF
        SS   OO      OO  RR    RR       II     NN      NN      TT      EE          RR    RR    FF          ....
        SS   OO      OO  RR    RR       II     NN      NN      TT      EE          RR    RR    FF          ....
SSSSSSSS      OOOOOO    RR      RR     IIIIII   NN      NN      TT      EEEEEEEEEE  RR      RR  FF          ....
SSSSSSSS      OOOOOO    RR      RR     IIIIII   NN      NN      TT      EEEEEEEEEE  RR      RR  FF          ....

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
```

```
;     1          0001  0 MODULE SOR$INTERFACE (
;     2          0002  0                 IDENT = 'V04-000')=     ! File: SORINTERF.B32 Edit: PDG3059
;     3          0003  1 BEGIN
;     4          0004  1 MACRO           IDENT = 'V03-059' %;
;     5          0005  1 !
;     6          0006  1 !**************************************************************************************
;     7          0007  1 !*                                                                                    *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                           *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                             *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                               *
;    11          0011  1 !*                                                                                    *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED            *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE           *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER           *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY           *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY           *
;    17          0017  1 !*  TRANSFERRED.                                                                       *
;    18          0018  1 !*                                                                                    *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE           *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT           *
;    21          0021  1 !*  CORPORATION.                                                                       *
;    22          0022  1 !*                                                                                    *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS           *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                           *
;    25          0025  1 !*                                                                                    *
;    26          0026  1 !*                                                                                    *
;    27          0027  1 !**************************************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 !
;    32          0032  1 ! FACILITY:     VAX-11 SORT/MERGE
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:
;    35          0035  1 !
;    36          0036  1 !       This module contains the user-visible interface routines to Sort/Merge.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:  VAX/VMS user mode
;    39          0039  1 !
;    40          0040  1 ! AUTHOR: Peter D Gilbert, CREATION DATE: 07-Jan-1982
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 !       T03-015         Original
;    45          0045  1 !       T03-019 Use COM_RDT_SIZ (instead of COM_FORMATS) to determine whether
;    46          0046  1 !               spec file processing is needed.  PDG 13-Dec-1982
;    47          0047  1 !       T03-020 Free COM_WRK_EFN.  If DDB_CHAN is non-zero, deass the channel.
;    48          0048  1 !               PDG 30-Dec-1982
;    49          0049  1 !       T03-021 Make work-file description blocks (WFBs) distinct from DDBs.
;    50          0050  1 !               PDG 31-Dec-1982
;    51          0051  1 !       T03-022 Change stat names to SOR$K_xxx.  PDG 3-Jan-1983
;    52          0052  1 !       T03-023 Add clean-up routines.  Make the deallocation routine global,
;    53          0053  1 !               and have it clear the address.  PDG 4-Jan-1983
;    54          0054  1 !       T03-024 Allow SOR$STATISTIC to be called at any time.  PDG 5-Jan-1983
;    55          0055  1 !       T03-025 New interface for collating sequence.  PDG 26-Jan-1983
;    56          0056  1 !       T03-026 Change WRK_USE to WRK_ALQ.  Allow stable option for merges.
;    57          0057  1 !               PDG 27-Jan-1983
```

```
;    58          0058  1 !       T03-027 Use COM_MERGE (rather than COM_MRG_ORDER) to indicate a merge.
;    59          0059  1 !               PDG 31-Jan-1983
;    60          0060  1 !       T03-028 Changes for hostile environment.  PDG 3-Feb-1983
;    61          0061  1 !       T03-029 Some fixes for asynchronous aborts.  PDG 4-Feb-1983
;    62          0062  1 !       T03-030 Call COLL$TIE_BREAK for DEC_MULTI.  PDG 16-Feb-1983
;    63          0063  1 !       T03-031 Complete asynchronous aborts.  PDG 28-Feb-1983
;    64          0064  1 !       T03-032 Changed name of SOR$STAT.  PDG 3-Mar-1983
;    65          0065  1 !       T03-033 Return worst status encountered.  PDG 7-Mar-1983
;    66          0066  1 !       T03-034 Remove need for COLL$COMPRESS.  PDG 5-Apr-1983
;    67          0067  1 !       T03-035 Give the SOR$RO_CODE__n PSECTs the EXE attr.  PDG 7-Apr-1983
;    68          0068  1 !       T03-036 SIGNAL option changed to NOSIGNAL.  PDG 11-Apr-1983
;    69          0069  1 !       T03-037 Reverse sense of tie-breaking only for DEC_MULTI.
;    70          0070  1 !               Information hiding of WFB structure.  PDG 12-Apr-1983
;    71          0071  1 !       T03-038 Fix access violation if END_SORT called when there's no
;    72          0072  1 !               context area for the sort/merge.  PDG 14-Apr-1983
;    73          0073  1 !       T03-039 Always reverse sense of tie-breaking.  PDG 18-Apr-1983
;    74          0074  1 !       T03-040 Recover from RMS$_RTB errors.  PDG 21-Apr-1983
;    75          0075  1 !       T03-041 Correctly set the COM_SIGNAL flag.  PDG 27-Apr-1983
;    76          0076  1 !       V03-042 Change ident.  PDG 28-Apr-1983
;    77          0077  1 !       V03-043 Make the SOR$$END_PSECTs long-relative.  PDG 2-Mar-1983
;    78          0078  1 !       V03-044 Improve code in the $PUT loop.  PDG 11-May-1983
;    79          0079  1 !       V03-045 Make RETURN_REC keep returning SS$_ENDOFFILE.  PDG 14-Jul-1983
;    80          0080  1 !       V03-046 Don't complain so much if LIB$FREE_VM fails.  PDG 31-Aug-1983
;    81          0081  1 !               Return useful STAT_K_INI_RUNS during dispersion phase.
;    82          0082  1 !       V03-047 Add support for checkpointing.  PDG 28-Sep-1983
;    83          0083  1 !       V03-048 Fix error in comparing worst severities.  PDG 14-Dec-1983
;    84          0084  1 !       V03-049 Check for specifying both NODUPS and STABLE.  PDG 16-Dec-1983
;    85          0085  1 !       V03-050 Make SOR_CONTEXT not writable if hostile.  PDG 16-Dec-1983
;    86          0086  1 !       V03-051 For hostile sorts, default the value of FILE_ALLOC here,
;    87          0087  1 !               since SOR$$OPEN is not called.  PDG 24-Jan-1984
;    88          0088  1 !       V03-052 Call SOR$$EBCDIC to initialize EBCDIC tables.  PDG 31-Jan-1984
;    89          0089  1 !       V03-053 Add COLL_DESC parameter to SOR$SPEC_FILE.  PDG 22-Feb-1984
;    90          0090  1 !       V03-054 Add comments on the calculation for TREE_INIT.  PDG 13-Mar-1984
;    91          0091  1 !       V03-055 Add VAXELN specific code.  Jeff East 1/2784
;    92          0092  1 !       V03-056 Add recovery from RMS$_RLK errors.  PDG 2-Apr-1984
;    93          0093  1 !       V03-057 Minor changes allocating buffers for MERGE.  PDG 9-Apr-1984
;    94          0094  1 !       V03-058 Set context longword back to zero within the CONTEXT_ macro
;    95          0095  1 !               if END_SORT is called with zero context.  PDG 15-May-1984
;    96          0096  1 !       V03-059 Change COM_RHB to COM_RHB_INP and COM_RHB_OUT.
;    97          0097  1 !               This is to avoid problems with merge, where an incoming
;    98          0098  1 !               record overwrites the VFC area for the outgoing record.
;    99          0099  1 !               PDG 24-Jul-1984
;   100          0100  1 !--
```

```
;  102          0101  1 LIBRARY 'SYS$LIBRARY:STARLET';
;  103          0102  1 REQUIRE 'SRC$:COM';
;  104          0172  1
;  105          0173  1 %IF NOT HOSTILE %THEN
;  106          0174  1 OWN
;  107          0175  1     SOR_CONTEXT:        REF BLOCK;              ! Default sort context info
;  108        U 0176  1 %ELSE
;  109        U 0177  1 BIND
;  110        U 0178  1     SOR_CONTEXT=        UPLIT(0);               ! Longword not writable
;  111          0179  1 %FI
;  112          0180  1
;  113          0181  1 LITERAL
;  114          0182  1     REG_INI_CTX = 3;
;  115          0183  1 LINKAGE
;  116          0184  1     AND_RETURN_R1 = CALL(;REGISTER=1),
;  117          0185  1     JSB_INI_CTX = JSB(REGISTER=REG_INI_CTX):
;  118          0186  1         NOPRESERVE(0,1,2,4,5)
;  119          0187  1         NOTUSED(6,7,8,9,10)
;  120          0188  1         GLOBAL(CTX=COM_REG_CTX);
;  121          0189  1
;  122          0190  1 FORWARD ROUTINE
;  123          0191  1     COND_HAND,                                  ! Handle exception conditions
;  124          0192  1     SOR$$ERROR,                                 ! Issue error diagnostic
;  125          0193  1     INI_CTX:            JSB_INI_CTX NOVALUE,    ! Initialize context area
;  126          0194  1     SOR$$ALLOCATE:      CAL_CTXREG,             ! Allocate storage
;  127          0195  1     SOR$$DEALLOCATE:    CAL_CTXREG NOVALUE,     ! Deallocate storage
;  128          0196  1     SOR$BEGIN_SORT,                             ! Initialize the sort
;  129          0197  1     SOR$SORT_MERGE,                             ! Sort or merge the data
;  130          0198  1     SOR$RELEASE_REC,                            ! Release record to sort/merge
;  131          0199  1     SOR$RETURN_REC,                             ! Return record from sort/merge
;  132          0200  1     AST_END_SORT:       JSB_INI_CTX,            ! Asynchronous termination
;  133          0201  1     SOR$END_SORT,                               ! Termination routine
;  134          0202  1     SOR$BEGIN_MERGE,                            ! Initialize the merge
;  135          0203  1     %IF NOT HOSTILE %THEN
;  136          0204  1     CLOSE_FILE:         CAL_CTXREG NOVALUE,     ! Close a file
;  137          0205  1     FREE_DDBS:          CAL_CTXREG NOVALUE,     ! Free storage for list of DDBs
;  138          0206  1     MRG_OUTPUT:         CAL_CTXREG NOVALUE,     ! Merge into the output file
;  139          0207  1     SOR$PASS_FILES:     AND_RETURN_R1,          ! Pass file information to sort
;  140          0208  1     SOR$SPEC_FILE,                              ! Process the specification file
;  141          0209  1     %FI
;  142          0210  1     SOR$STAT;                                   ! Get a statistic
;  143          0211  1
;  144          0212  1 LITERAL
;  145          0213  1     SOR_K_END_AST =     TRUE;   ! Indicates SOR$END_SORT should work with ASTs
;  146          0214  1
;  147        L 0215  1 %IF HOSTILE
;  148        U 0216  1 %THEN
;  149        U 0217  1     MACRO
;  150        U 0218  1         LIB$GET_VM = SOR$LIB$GET_VM %,
;  151        U 0219  1         LIB$FREE_VM = SOR$LIB$FREE_VM %,
;  152        U 0220  1         LIB$SIGNAL = SOR$LIB$SIGNAL %,
;  153        U 0221  1         LIB$AST_IN_PROG = SOR$LIB$AST_IN_PROG %,
;  154        U 0222  1         SYS$CLRAST = SOR$SYS$CLRAST %,
;  155        U 0223  1         SYS$DCLAST = SOR$SYS$DCLAST %,
;  156        U 0224  1         SYS$UNWIND = SOR$SYS$UNWIND %,
;  157        U 0225  1         SYS$ADJWSL = SOR$SYS$ADJWSL %;
;  158          0226  1 %FI
```

```
;  160          0227  1 ! Define user-visible literals
;  161          0228  1 !
;  162          0229  1 MACRO
;  163          0230  1     FNV_(O,P,S,E) = P %,
;  164          0231  1     FNM_(O,P,S,E) = 1^P %,
;  165          0232  1     DFX_(A,B,C)[D] = %NAME(A,D) = C (%NAME(B,D)) %;
;  166          0233  1 MACRO
;  167        M 0234  1     OPTION_FIELDS =
;  168        M 0235  1         'EBCDIC',
;  169        M 0236  1 !       'FIXED',
;  170        M 0237  1         'MULTI',
;  171        M 0238  1         'NODUPS',
;  172        M 0239  1         'NOSIGNAL',
;  173        M 0240  1         'SEQ_CHECK',
;  174        M 0241  1         'STABLE',
;  175          0242  1         'LOAD_FILL' %;
;  176          0243  1
;  177          0244  1 GLOBAL LITERAL
;  178        P 0245  1     DFX_('SOR$GK_', 'TYP_K_',
;  179          0246  1         'RECORD', 'TAG', 'INDEX', 'ADDRESS'),
;  180          0247  1     DFX_('SOR$M_', 'OPT_', %QUOTE FNM_, OPTION_FIELDS),
;  181          0248  1     DFX_('SOR$V_', 'OPT_', %QUOTE FNV_, OPTION_FIELDS),
;  182        P 0249  1     DFX_('SOR$K_', 'STAT_K_',
;  183        P 0250  1         'MAX_STAT', 'IDENT', 'REC_INP', 'REC_SOR', 'REC_OUT', 'LRL_INP',
;  184        P 0251  1         'LRL_INT',
;  185        P 0252  1         'LRL_OUT', 'NODES', 'INI_RUNS', 'MRG_ORDER', 'MRG_PASSES', 'WRK_ALQ',
;  186          0253  1         'MBC_INP', 'MBC_OUT', 'MBF_INP', 'MBF_OUT');
```

```
;  188          0254  1 ! Macros to test for optional parameters.
;  189          0255  1 !
;  190          0256  1 !   FIRSTPARAMETER_
;  191          0257  1 !       Define the first parameter, for use by PRESENT_ and NULL_.
;  192          0258  1 !
;  193          0259  1 !   PRESENT_
;  194          0260  1 !       Test for a parameter present.
;  195          0261  1 !
;  196          0262  1 !   NULL_
;  197          0263  1 !       Test for a parameter present, and whether it equals zero.
;  198          0264  1 !
;  199          0265  1 MACRO
;  200        M 0266  1     PRESENT_(X) =
;  201        M 0267  1         BEGIN
;  202        M 0268  1         BUILTIN ACTUALCOUNT;
;  203        M 0269  1         LITERAL Y__ = 1+(X-FIRSTPARAMETER__)/%UPVAL;
;  204        M 0270  1         ACTUALCOUNT() GEQU Y__
;  205          0271  1         END %,
;  206        M 0272  1     NULL_(X) =
;  207        M 0273  1         BEGIN
;  208        M 0274  1         BUILTIN NULLPARAMETER;
;  209        M 0275  1         LITERAL Y__ = 1+(X-FIRSTPARAMETER__)/%UPVAL;
;  210        M 0276  1         NULLPARAMETER(Y__)
;  211          0277  1         END %,
;  212        M 0278  1     FIRSTPARAMETER_(X) =
;  213          0279  1         MACRO FIRSTPARAMETER__ = X %QUOTE % %;
```

```
;   215          0280  1 ! Macro to return status from the user-visible routines
;   216          0281  1 ! Note that all successful returns from the user-visible routines
;   217          0282  1 ! (except END_SORT and STAT) should use this macro, passing SS$_NORMAL.
;   218          0283  1 !
;   219          0284  1 MACRO
;   220        M 0285  1     RETURN__(X) =
;   221        M 0286  1         BEGIN
;   222        M 0287  1         %IF SOR_K_END_AST %THEN CTX__[0] = .CTX__[0] AND NOT %B'1'; %FI
;   223        M 0288  1         %IF %IDENTICAL(X, SS$_NORMAL)
;   224        M 0289  1         %THEN
;   225        M 0290  1             CTX[COM_FLO_ABORT] = FALSE;
;   226        M 0291  1             RETURN .CTX[COM_WORST]
;   227        M 0292  1         %ELSE
;   228        M 0293  1             RETURN X
;   229        M 0294  1         %FI
;   230          0295  1         END %;
;   231          0296  1
;   232          0297  1 ! Macro to get the address of common context information.
;   233          0298  1 !
;   234          0299  1 MACRO
;   235        M 0300  1     CONTEXT(CONTEXT,ERR) =
;   236        M 0301  1         GLOBAL REGISTER
;   237        M 0302  1             CTX = COM_REG_CTX:  REF CTX_BLOCK;
;   238        M 0303  1         LOCAL
;   239        M 0304  1             CTX__:      REF VECTOR[1] VOLATILE; ! Address of context longword
;   240        M 0305  1         ENABLE
;   241        M 0306  1             COND_HAND(CTX__);
;   242        M 0307  1         BEGIN
;   243        M 0308  1         BUILTIN
;   244        M 0309  1             TESTBITSS;
;   245        M 0310  1         REGISTER
;   246        M 0311  1             P__ = REG_INI_CTX:  REF VECTOR[1];  ! Address of context pointer
;   247        M 0312  1         MACRO
;   248        M 0313  1             IS_END_SORT = NOT %NULL(ERR) AND ERR+0 %QUOTE %,
;   249        M 0314  1             IS_STAT     = NOT %DECLARED(COM_FLO_ABORT) %QUOTE %;
;   250        M 0315  1         !
;   251        M 0316  1         ! Get the address to store the address of the context area.
;   252        M 0317  1         ! If the context parameter is missing, use the default context pointer.
;   253        M 0318  1         ! If present and non-zero, use it.
;   254        M 0319  1         ! If present and zero, use the default context pointer.
;   255        M 0320  1         !
;   256        M 0321  1         %IF NOT HOSTILE
;   257        M 0322  1         %THEN
;   258        M 0323  1             IF NOT PRESENT(CONTEXT) THEN P__ = SOR_CONTEXT
;   259        M 0324  1             ELIF (P__ = .CONTEXT) NEQ 0 THEN 0
;   260        M 0325  1             ELSE P__ = SOR_CONTEXT;
;   261        M 0326  1         %ELSE
;   262        M 0327  1             P__ = .CONTEXT;
;   263        M 0328  1         %FI
;   264        M 0329  1         %IF NOT IS_STAT
;   265        M 0330  1         %THEN
;   266        M 0331  1             CTX__ = P__[0];
;   267        M 0332  1         %FI
;   268        M 0333  1         !
;   269        M 0334  1         ! Test and set the synchronization bit to see whether any other
;   270        M 0335  1         ! routines for this sort/merge are currently active.
;   271        M 0336  1         !
```

```
;  272          M 0337  1          %IF SOR_K_END_AST AND NOT IS_STAT
;  273          M 0338  1          %THEN
;  274          M 0339  1              IF TESTBITSS(P__[0])
;  275          M 0340  1              THEN
;  276          M 0341  1                  %IF IS_END_SORT
;  277          M 0342  1                  %THEN
;  278          M 0343  1                      RETURN AST_END_SORT(P__[0])
;  279          M 0344  1                  %ELSE
;  280          M 0345  1                      ! Only END_SORT and STAT may be called asynchronously.
;  281          M 0346  1                      !
;  282          M 0347  1                      RETURN SOR$_SORT_ON
;  283          M 0348  1                  %FI;
;  284          M 0349  1          %FI
;  285          M 0350  1          !
;  286          M 0351  1          ! Fetch the address of the context area
;  287          M 0352  1          !
;  288          M 0353  1          IF (CTX = .P__[0] AND NOT %B'1') EQL 0
;  289          M 0354  1          THEN
;  290          M 0355  1              %IF %NULL(ERR)
;  291          M 0356  1              %THEN
;  292          M 0357  1                  BEGIN
;  293          M 0358  1                  !
;  294          M 0359  1                  ! Allocate and initialize the context area
;  295          M 0360  1                  !
;  296          M 0361  1                  INI_CTX(P__[0]);
;  297          M 0362  1                  END
;  298          M 0363  1              %ELSE
;  299          M 0364  1                  BEGIN
;  300          M 0365  1                  !
;  301          M 0366  1                  ! This routine does not initialize the context area
;  302          M 0367  1                  !
;  303          M 0368  1                  %IF IS_END_SORT %THEN
;  304          M 0369  1                      CTX__[0] = 0;
;  305          M 0370  1                  %FI
;  306          M 0371  1                  %IF IS_STAT OR IS_END_SORT %THEN
;  307          M 0372  1                      RETURN ERR
;  308          M 0373  1                  %ELSE
;  309          M 0374  1                      RETURN__(ERR)
;  310          M 0375  1                  %FI
;  311          M 0376  1                  END
;  312          M 0377  1              %FI;
;  313          M 0378  1          !
;  314          M 0379  1          ! Test and set a bit indicating that this sort is only fit
;  315          M 0380  1          ! to be aborted.
;  316          M 0381  1          !
;  317          M 0382  1          %IF IS_END_SORT %THEN
;  318          M 0383  1              CTX[COM_FLO_ABORT] = TRUE
;  319          M 0384  1          %ELSE %IF IS_STAT %THEN
;  320          M 0385  1              0
;  321          M 0386  1          %ELSE
;  322          M 0387  1              IF TESTBITSS(CTX[COM_FLO_ABORT])
;  323          M 0388  1              THEN
;  324          M 0389  1                  RETURN__(SOR$_SORT_ON)
;  325          M 0390  1          %FI %FI;
;  326          M 0391  1          CTX[COM_WORST] = SS$_NORMAL;
;  327            0392  1          END %;
;  328            0393  1
```

```
;   329          0394  1 ! This macro causes a checkpoint to be taken when needed
;   330          0395  1 !
;   331          0396  1 MACRO
;   332        M 0397  1     CHECKPOINT_COUNTDOWN( DUMMY ) =
;   333        M 0398  1         %IF FUN_K_CHECKPOINT
;   334        M 0399  1         %THEN
;   335        M 0400  1             BEGIN
;   336        M 0401  1             %IF NOT %DECLARED(R0) %THEN BUILTIN R0; %FI
;   337        M 0402  1             IF (CTX[COM_COUNTDOWN] = .CTX[COM_COUNTDOWN] - 1) LSS 0
;   338        M 0403  1             THEN
;   339        M 0404  1                 R0 = SOR$$CHECKPOINT(); ! This routine is really NOVALUE
;   340        M 0405  1             END
;   341        M 0406  1         %ELSE
;   342        M 0407  1             0
;   343          0408  1         %FI %;
```

```
;  345          0409  1 EXTERNAL ROUTINE
;  346          0410  1     SOR$$KEY_SUB:        CAL_CTXREG,                ! Process keys
;  347          0411  1     SOR$$TREE_INIT:      CAL_CTXREG,                ! Allocate and initialize tree
;  348          0412  1     SOR$$TREE_INSERT:    JSB_INSERT,                ! Insert record in sort tree
;  349          0413  1     SOR$$TREE_EXTRACT:   JSB_EXTRACT,               ! Get a record from sort tree
;  350          0414  1     SOR$$WRK_ALQ:        CAL_CTXREG,                ! Calculate work file allocation
;  351        U 0415  1 %IF FUN_K_CHECKPOINT %THEN
;  352        U 0416  1     SOR$$CHECKPOINT:     CAL_CTXREG,                ! Take a checkpoint
;  353          0417  1 %FI
;  354        L 0418  1 %IF NOT HOSTILE
;  355          0419  1 %THEN
;  356          0420  1     SOR$$SPEC_KEY_SUB:   CAL_CTXREG,                ! Process keys from spec file
;  357          0421  1     SOR$$SPEC_FILE:      CAL_CTXREG,                ! Process specification text
;  358          0422  1     SOR$$OPEN:           CAL_CTXREG,                ! Open input and output files
;  359          0423  1     COLL$INIT,                                      ! Initialize collating sequence
;  360          0424  1     SOR$$DECM:           CAL_CTXREG,                ! Initialize to DEC_MULTI
;  361          0425  1     SOR$$EBCDIC:         CAL_CTXREG,                ! Initialize to EBCDIC
;  362          0426  1     COLL$BASE,                                      ! Define the base collating sequence
;  363          0427  1     COLL$PAD,                                       ! Specify pad character
;  364          0428  1     COLL$TIE_BREAK,                                 ! indicate tie-breaking
;  365          0429  1     COLL$RESULT,                                    ! Convert result to code
;  366          0430  1     STR$APPEND:          ADDRESSING_MODE(GENERAL),      ! Append strings
;  367          0431  1     SOR$$COPY_FILE_NAME:CAL_CTXREG NOVALUE,
;  368          0432  1 %FI
;  369          0433  1     SOR$$FREE_FILE_NAME:CAL_CTXREG NOVALUE,
;  370          0434  1     LIB$GET_VM:          ADDRESSING_MODE(GENERAL),      ! Get virtual memory
;  371          0435  1     LIB$FREE_VM:         ADDRESSING_MODE(GENERAL),      ! Free virtual memory
;  372          0436  1     LIB$SIGNAL:          ADDRESSING_MODE(GENERAL);      ! Signal errors
```

```
;  374     0437  1 ROUTINE COND_HAND
;  375     0438  1         (
;  376     0439  1         SIGVEC: REF BLOCK[,BYTE],       ! Signal vector
;  377     0440  1         MCHVEC: REF BLOCK[,BYTE],       ! Mechanism vector
;  378     0441  1         ENAVEC: REF VECTOR              ! Enable vector
;  379     0442  1         ) =
;  380     0443  1 !++
;  381     0444  1 !
;  382     0445  1 ! FUNCTIONAL DESCRIPTION:
;  383     0446  1 !
;  384     0447  1 !       Condition handler for errors occurring during sort or merge.
;  385     0448  1 !       This routine checks whether errors should be returned or signalled.
;  386     0449  1 !       If errors are to be signalled, we resignal the error.
;  387     0450  1 !       If error statuses are to be returned then
;  388     0451  1 !           If the error is only a warning or informational, just continue
;  389     0452  1 !           Otherwise, we unwind to the caller.
;  390     0453  1 !
;  391     0454  1 !       Before modifying this routine, consider how reserved opcode or operand
;  392     0455  1 !       errors will be delivered (users frequently want to catch these).
;  393     0456  1 !
;  394     0457  1 ! FORMAL PARAMETERS:
;  395     0458  1 !
;  396     0459  1 !       SIGVEC.ra.r             The signal vector
;  397     0460  1 !       MCHVEC.ra.r             The mechanism vector
;  398     0461  1 !       ENAVEC.ra.r             The enable vector
;  399     0462  1 !
;  400     0463  1 ! IMPLICIT INPUTS:
;  401     0464  1 !
;  402     0465  1 !       NONE
;  403     0466  1 !
;  404     0467  1 ! IMPLICIT OUTPUTS:
;  405     0468  1 !
;  406     0469  1 !       NONE
;  407     0470  1 !
;  408     0471  1 ! ROUTINE VALUE:
;  409     0472  1 !
;  410     0473  1 !       Status code.
;  411     0474  1 !
;  412     0475  1 ! SIDE EFFECTS:
;  413     0476  1 !
;  414     0477  1 !       NONE
;  415     0478  1 !
;  416     0479  1 !--
;  417     0480  2     BEGIN
;  418     0481  2     LOCAL
;  419     0482  2         CTX:    REF CTX_BLOCK,          ! Address of context area
;  420     0483  2         CAD:    REF VECTOR[1];          ! Address of context longword
;  421     0484  2
;  422     0485  2     BUILTIN
;  423     0486  2         NULLPARAMETER,
;  424     0487  2         AP,
;  425     0488  2         CALLG;
;  426     0489  2
;  427     0490  2     BIND
;  428     0491  2         SIG_NAME = SIGVEC[CHF$L_SIG_NAME]: BLOCK[,BYTE];
;  429     0492  2
;  430     0493  2     ! If we can't find the context area, resignal the error.
```

```
;  431          0494  2      !
;  432          0495  2      IF
;  433          0496  2          BEGIN
;  434          0497  2          !
;  435          0498  2          ! ENAVEC[1] contains the address of a longword.
;  436          0499  2          ! This longword holds the address of the user's context longword.
;  437          0500  2          ! Thus, ..ENAVEC[1] is the address of the user's context longword,
;  438          0501  2          ! And  ...ENAVEC[1] is the address of the context area.
;  439          0502  2          !
;  440          0503  2          IF NULLPARAMETER(3) THEN TRUE
;  441          0504  2          ELIF (CAD = .ENAVEC[1]) EQL 0 THEN TRUE
;  442          0505  2          ELIF (CAD = .CAD[0]) EQL 0 THEN TRUE
;  443          0506  2          ELIF (CTX = .CAD[0] AND NOT %B'1') EQL 0 THEN TRUE
;  444          0507  2          ELSE FALSE
;  445          0508  2          END
;  446          0509  2      THEN
;  447          0510  2          BEGIN
;  448          0511  2          SIG_NAME[STS$V_SEVERITY] = STS$K_SEVERE;
;  449          0512  2          RETURN SS$_RESIGNAL;                ! No context, resignal
;  450          0513  2          END
;  451          0514  2      ELIF .SIG_NAME EQL SS$_UNWIND
;  452          0515  2      THEN
;  453          0516  2          BEGIN
;  454          0517  2          !
;  455          0518  2          ! If we are unwinding, indicate that the only valid routine to call is
;  456          0519  2          ! END_SORT, indicate that we are no longer active, and return.
;  457          0520  2          !
;  458          0521  2          CTX[COM_FLO_ABORT] = TRUE;
;  459          0522  2          CAD[0] = .CAD[0] AND NOT %B'1';
;  460          0523  2          RETURN SS$_RESIGNAL;
;  461          0524  2          END
;  462        L 0525  2      %IF SOR_K_END_AST
;  463          0526  2      %THEN
;  464          0527  2      ELIF
;  465          0528  2          .SIG_NAME EQL SOR$_END_SORT
;  466          0529  2      THEN
;  467          0530  3          BEGIN
;  468          0531  3          IF
;  469          0532  3              .MCHVEC[CHF$L_MCH_DEPTH] EQL 0      ! Did we signal this?
;  470          0533  3          THEN
;  471          0534  3              RETURN SS$_RESIGNAL
;  472          0535  3          ELIF
;  473          0536  4              BEGIN                               ! Request to unwind?
;  474          0537  4              BIND SIG_VL = SIGVEC: REF VECTOR[,LONG];
;  475          0538  4              (.SIGVEC[CHF$L_SIG_ARGS] GEQU 5) AND
;  476          0539  4              (.SIG_VL[2] GTR 0) AND
;  477          0540  5              (.SIG_VL[3] EQL CAD[0])
;  478          0541  4              END
;  479          0542  3          THEN
;  480          0543  4              BEGIN
;  481          0544  4              !
;  482          0545  4              ! Unwind to a call to SOR$END_SORT, and return an abort message.
;  483          0546  4              !
;  484          0547  4              LINKAGE
;  485          0548  4                  LINK_R1 = CALL(REGISTER=1);
;  486          0549  4              ROUTINE END_SORT(CTXADR: REF VECTOR[1]): LINK_R1 =
;  487          0550  5                  BEGIN
```

```
;  488          0551  5     ESTABLISH_(0);                          ! No condition handler
;  489          0552  5     CTXADR[0]= .CTXADR[0] AND NOT %B'1';
;  490          0553  5     SOR$END_SORT(CTXADR[0]);                ! End the sort/merge
;  491          0554  5     RETURN SOR$_END_SORT AND NOT STS$M_SEVERITY OR STS$K_SEVERE;
;  492          0555  4     END;


                                        .TITLE  SOR$INTERFACE
                                        .IDENT  \V04-000\

                                        .PSECT  SOR$RW_PICDATA,NOEXE,  PIC,2

                                00000 SOR_CONTEXT:
                                        .BLKB   4

                                     SOR$GK_RECORD==     1
                                     SOR$GK_TAG==        2
                                     SOR$GK_INDEX==      3
                                     SOR$GK_ADDRESS==    4
                                     SOR$M_EBCDIC==      2
                                     SOR$M_MULTI==       4
                                     SOR$M_NODUPS==      64
                                     SOR$M_NOSIGNAL==    8
                                     SOR$M_SEQ_CHECK==   16
                                     SOR$M_STABLE==      1
                                     SOR$M_LOAD_FILL==   512
                                     SOR$V_EBCDIC==      1
                                     SOR$V_MULTI==       2
                                     SOR$V_NODUPS==      6
                                     SOR$V_NOSIGNAL==    3
                                     SOR$V_SEQ_CHECK==   4
                                     SOR$V_STABLE==      0
                                     SOR$V_LOAD_FILL==   9
                                     SOR$K_MAX_STAT==    16
                                     SOR$K_IDENT==       0
                                     SOR$K_REC_INP==     1
                                     SOR$K_REC_SOR==     2
                                     SOR$K_REC_OUT==     3
                                     SOR$K_LRL_INP==     4
                                     SOR$K_LRL_INT==     5
                                     SOR$K_LRL_OUT==     6
                                     SOR$K_NODES==       7
                                     SOR$K_INI_RUNS==    8
                                     SOR$K_MRG_ORDER==   9
                                     SOR$K_MRG_PASSES==  10
                                     SOR$K_WRK_ALQ==     11
                                     SOR$K_MBC_INP==     12
                                     SOR$K_MBC_OUT==     13
                                     SOR$K_MBF_INP==     14
                                     SOR$K_MBF_OUT==     15
                                        .EXTRN  SOR$$KEY_SUB, SOR$$TREE_INIT
                                        .EXTRN  SOR$$TREE_INSERT
                                        .EXTRN  SOR$$TREE_EXTRACT
                                        .EXTRN  SOR$$WRK_ALQ, SOR$$SPEC_KEY_SUB
                                        .EXTRN  SOR$$SPEC_FILE, SOR$$OPEN
                                        .EXTRN  COLL$INIT, SOR$$DECM
                                        .EXTRN  SOR$$EBCDIC, COLL$BASE
```

```
                                                        .EXTRN  COLL$PAD, COLL$TIE_BREAK
                                                        .EXTRN  COLL$RESULT, STR$APPEND
                                                        .EXTRN  SOR$$COPY_FILE_NAME
                                                        .EXTRN  SOR$$FREE_FILE_NAME
                                                        .EXTRN  LIB$GET_VM, LIB$FREE_VM
                                                        .EXTRN  LIB$SIGNAL

                                                        .PSECT  SOR$RO_CODE,NOWRT,  SHR,  PIC,2

                                            0000 00000 END_SORT:
                                                        .WORD   Save nothing                            ; 0549
                                        6D  D4 00002            CLRL    (FP)                            ; 0551
                                        01  8A 00004            BICB2   #1, (CTXADR)                    ; 0552
                                61      51  DD 00007            PUSHL   CTXADR                          ; 0553
                            0000V   CF  01  FB 00009            CALLS   #1, SOR$END_SORT                ;
                                50 001C80EC 8F  D0 0000E        MOVL    #1868012, R0                    ; 0554
                                            04 00015            RET                                     ; 0555

; Routine Size:  22 bytes,    Routine Base:  SOR$RO_CODE + 0000

;  493          0556  4              CTX[COM_FLO_ABORT] = TRUE;
;  494          0557  4              MCHVEC[CHF$L_MCH_SAVR1] = CAD[0];   ! Addr of context longword
;  495          0558  4              RETURN $UNWIND(DEPADR=MCHVEC[CHF$L_MCH_DEPTH], NEWPC=END_SORT+2);
;  496          0559  4              END
;  497          0560  3          ELSE
;  498          0561  3              RETURN SS$_RESIGNAL;
;  499          0562  3          END
;  500          0563  3      %FI
;  501          0564  2      ELIF .SIG_NAME[STS$V_SEVERITY] GEQU STS$K_SEVERE
;  502          0565  2      THEN
;  503          0566  3          BEGIN
;  504          0567  3          CTX[COM_FLO_ABORT] = TRUE;              ! May only call END_SORT
;  505          0568  3          IF .CTX[COM_SIGNAL]
;  506          0569  3          THEN
;  507          0570  3              RETURN SS$_RESIGNAL;                ! Signal errors
;  508          0571  3          MCHVEC[CHF$L_MCH_SAVR0] = .SIG_NAME;
;  509          0572  3          RETURN $UNWIND();                       ! Return error status to caller
;  510          0573  3          END
;  511          0574  2      ELSE
;  512          0575  3          BEGIN
;  513          0576  3          !
;  514          0577  3          ! Update the worst error we've seen, and either resignal or continue.
;  515          0578  3          ! If we resignal, make the worst error an informational,
;  516          0579  3          !       since the user will have already seen it.
;  517          0580  3          ! If we continue, make the worst error the same as the signalled error,
;  518          0581  3          !       to make the user more aware of what he missed.
;  519          0582  3          !
;  520          0583  3          IF .CTX[COM_SIGNAL]
;  521          0584  3          THEN
;  522          0585  4              BEGIN
;  523          0586  4              IF .SIG_NAME[STS$V_FAC_NO] EQL SORT$_FACILITY
;  524          0587  4              THEN
;  525          0588  4                  CTX[COM_WORST] = SOR$_ENDDIAGS AND NOT STS$M_SEVERITY OR
;  526          0589  4                      STS$K_INFO;                 ! Just an informational
;  527          0590  4              RETURN SS$_RESIGNAL;                ! Signal errors
;  528          0591  4              END
```

```
;  529        0592  3          ELSE
;  530        0593  4              BEGIN
;  531        0594  4              IF .SIG_NAME[STS$V_FAC_NO] EQL SORT$_FACILITY
;  532        0595  4              THEN
;  533        0596  5                  BEGIN
;  534        0597  5                  BIND
;  535        0598  5                      SEV = UPLIT BYTE(2,0,3,1,4,4,4,4): VECTOR[,BYTE];
;  536        0599  5                  IF .SEV[.SIG_NAME AND 7] GTRU .SEV[.CTX[COM_WORST] AND 7]
;  537        0600  5                  THEN
;  538        0601  5                      CTX[COM_WORST] = .SIG_NAME;
;  539        0602  5                  CTX[COM_WORST] = .CTX[COM_WORST] AND STS$M_SEVERITY
;  540        0603  5                      OR SOR$_ENDDIAGS AND NOT STS$M_SEVERITY;
;  541        0604  4                  END;
;  542        0605  4              RETURN SS$_CONTINUE;                        ! Ignore trivial errors
;  543        0606  3              END;
;  544        0607  2          END;
;  545        0608  2
;  546        0609  1      END;


                    04  04  04  04  01  03  00  02  00016 P.AAA:  .BYTE   2, 0, 3, 1, 4, 4, 4, 4                    ;

                                                          SEV=                P.AAA
                                                                  .EXTRN  SYS$UNWIND

                                                  003C 00000 COND_HAND:
                                                                  .WORD   Save R2,R3,R4,R5                          ; 0437
                                55        F3   AF  9E 00002       MOVAB   SEV, R5                                   ;
                                52        04   AC  D0 00006       MOVL    SIGVEC, R2                                ; 0491
                                54        04   A2  9E 0000A       MOVAB   4(R2), R4                                 ;
                                03             6C  91 0000E       CMPB    (AP), #3                                  ; 0503
                                               1A  1F 00011       BLSSU   1$                                        ;
                                          0C   AC  D5 00013       TSTL    12(AP)                                    ;
                                               15  13 00016       BEQL    1$                                        ;
                                50        0C   AC  D0 00018       MOVL    ENAVEC, R0                                ; 0504
                                53        04   A0  D0 0001C       MOVL    4(R0), CAD                                ;
                                               0B  13 00020       BEQL    1$                                        ;
                                53             63  D0 00022       MOVL    (CAD), CAD                                ; 0505
                                               06  13 00025       BEQL    1$                                        ;
                    50          63             01  CB 00027       BICL3   #1, (CAD), CTX                            ; 0506
                                               07  12 0002B       BNEQ    2$                                        ;
        64          03          00             04  F0 0002D 1$:   INSV    #4, #0, #3, (R4)                          ; 0511
                                               7D  11 00032       BRB     7$                                        ; 0513
                    00000920    8F             64  D1 00034 2$:   CMPL    (R4), #2336                               ; 0514
                                               09  12 0003B       BNEQ    3$                                        ;
                          5C    A0             10  88 0003D       BISB2   #16, 92(CTX)                              ; 0521
                                63             01  8A 00041       BICB2   #1, (CAD)                                 ; 0522
                                               6B  11 00044       BRB     7$                                        ; 0523
                    001C80E9    8F             64  D1 00046 3$:   CMPL    (R4), #1868009                            ; 0528
                                               2A  12 0004D       BNEQ    4$                                        ;
                                51        08   AC  D0 0004F       MOVL    MCHVEC, R1                                ; 0532
                                          08   A1  D5 00053       TSTL    8(R1)                                     ;
                                               59  13 00056       BEQL    7$                                        ;
                                05             62  D1 00058       CMPL    (R2), #5                                  ; 0538
                                               54  1F 0005B       BLSSU   7$                                        ;
                                          08   A2  D5 0005D       TSTL    8(R2)                                     ; 0539
```

```
                                                 4F  15 00060          BLEQ    7$                                      ;
                                          53      0C  A2  D1 00062          CMPL    12(R2), CAD                             ; 0540
                                                 49  12 00066          BNEQ    7$                                      ;
                                       5C  A0        10  88 00068          BISB2   #16, 92(CTX)                            ; 0556
                                       10  A1        53  D0 0006C          MOVL    CAD, 16(R1)                             ; 0557
                                             FF70    CF  9F 00070          PUSHAB  END_SORT+2                              ; 0558
                                               08    A1  9F 00074          PUSHAB  8(RT)                                   ;
                                                     1A  11 00077          BRB     5$                                      ;
                04          64          03          00  ED 00079 4$:      CMPZV   #0, #3, (R4), #4                        ; 0564
                                                     1B  1F 0007E          BLSSU   6$                                      ;
                                       5C  A0        10  88 00080          BISB2   #16, 92(CTX)                            ; 0567
                                   28  5B  A0        02  E0 00084          BBS     #2, 91(CTX), 7$                         ; 0568
                                           51    08  AC  D0 00089          MOVL    MCHVEC, R1                              ; 0571
                                       0C  A1        64  D0 0008D          MOVL    (R4), 12(R1)                            ;
                                                     7E  7C 00091          CLRQ    -(SP)                                   ; 0572
                        00000000G      00          02  FB 00093 5$:      CALLS   #2, SYS$UNWIND                          ;
                                                     04  0009A          RET                                             ;
                17      5B  A0        02  E1 0009B 6$:      BBC     #2, 91(CTX), 8$                         ; 0583
                1C  02  A4            0C          00  ED 000A0          CMPZV   #0, #12, 2(R4), #28                     ; 0586
                                                     09  12 000A6          BNEQ    7$                                      ;
                                 0130  C0 001C828B    8F  D0 000A8          MOVL    #1868427, 304(CTX)                      ; 0588
                                       50    0918    8F  3C 000B1 7$:      MOVZWL  #2328, R0                               ; 0593
                                                     04  000B6          RET                                             ;
                1C  02  A4            0C          00  ED 000B7 8$:      CMPZV   #0, #12, 2(R4), #28                     ; 0594
                                                     26  12 000BD          BNEQ    10$                                     ;
                51          64          03          00  EF 000BF          EXTZV   #0, #3, (R4), R1                        ; 0599
                                         53    0130  C0  9E 000C4          MOVAB   304(CTX), R3                            ;
                50          63          03          00  EF 000C9          EXTZV   #0, #3, (R3), R0                        ;
                                     6540            6541  91 000CE          CMPB    SEV[R1], SEV[R0]                        ;
                                                     03  1B 000D3          BLEQU   9$                                      ;
                                          63         64  D0 000D5          MOVL    (R4), (R3)                              ; 0601
                    50          63          03          00  EF 000D8 9$:      EXTZV   #0, #3, (R3), R0                        ; 0603
                                63          50 001C8288  8F  C9 000DD          BISL3   #1868424, R0, (R3)                      ;
                                            50          01  D0 000E5 10$:      MOVL    #1, R0                                  ; 0605
                                                        04  000E8          RET                                             ; 0609

; Routine Size:  233 bytes,    Routine Base:  SOR$RO_CODE + 001E
```

```
;  548          0610  1 GLOBAL ROUTINE SOR$$ERROR(ERR) =
;  549          0611  1
;  550          0612  1 !++
;  551          0613  1 !
;  552          0614  1 ! FUNCTIONAL DESCRIPTION:
;  553          0615  1 !
;  554          0616  1 !       This routine signals an error diagnostic.
;  555          0617  1 !
;  556          0618  1 ! FORMAL PARAMETERS:
;  557          0619  1 !
;  558          0620  1 !       Parameters passed to LIB$SIGNAL.
;  559          0621  1 !
;  560          0622  1 ! IMPLICIT INPUTS:
;  561          0623  1 !
;  562          0624  1 !       NONE
;  563          0625  1 !
;  564          0626  1 ! IMPLICIT OUTPUTS:
;  565          0627  1 !
;  566          0628  1 !       NONE
;  567          0629  1 !
;  568          0630  1 ! ROUTINE VALUE:
;  569          0631  1 !
;  570          0632  1 !       System status (first parameter of signalled status), with the
;  571          0633  1 !       INHIB_MSG bit set.
;  572          0634  1 !
;  573          0635  1 ! SIDE EFFECTS:
;  574          0636  1 !
;  575          0637  1 !       The image may be exitted due to the error.
;  576          0638  1 !
;  577          0639  1 !--
;  578          0640  2     BEGIN
;  579          0641  2     BUILTIN
;  580          0642  2         AP,
;  581          0643  2         CALLG;
;  582          0644  2     CALLG(.AP, LIB$SIGNAL);
;  583          0645  2     RETURN .ERR OR STS$M_INHIB_MSG;
;  584          0646  1     END;


                                                0000 00000          .ENTRY  SOR$$ERROR, Save nothing                ; 0610
                    00000000G   00              6C   FA 00002      CALLG   (AP), LIB$SIGNAL                        ; 0644
                         50       04   AC 10000000   8F   C9 00009  BISL3   #268435456, ERR, R0                   ; 0645
                                                     04 00012      RET                                             ; 0646

; Routine Size:  19 bytes,    Routine Base:  SOR$RO_CODE + 0107
```

```
;  586          0647  1 ROUTINE INI_CTX
;  587          0648  1         (
;  588          0649  1         CTXADR: REF VECTOR[1]
;  589          0650  1         ): JSB_INI_CTX NOVALUE =
;  590          0651  1
;  591          0652  1 !++
;  592          0653  1 !
;  593          0654  1 ! FUNCTIONAL DESCRIPTION:
;  594          0655  1 !
;  595          0656  1 !       This routine allocates and initializes the context area.
;  596          0657  1 !
;  597          0658  1 ! FORMAL PARAMETERS:
;  598          0659  1 !
;  599          0660  1 !       NONE
;  600          0661  1 !
;  601          0662  1 ! IMPLICIT INPUTS:
;  602          0663  1 !
;  603          0664  1 !       NONE
;  604          0665  1 !
;  605          0666  1 ! IMPLICIT OUTPUTS:
;  606          0667  1 !
;  607          0668  1 !       The process region may be expanded.
;  608          0669  1 !
;  609          0670  1 ! ROUTINE VALUE:
;  610          0671  1 !
;  611          0672  1 !       Address of the storage.
;  612          0673  1 !
;  613          0674  1 ! SIDE EFFECTS:
;  614          0675  1 !
;  615          0676  1 !       NONE
;  616          0677  1 !
;  617          0678  1 !--
;  618          0679  2     BEGIN
;  619          0680  2     EXTERNAL REGISTER
;  620          0681  2         CTX =   COM_REG_CTX:    REF CTX_BLOCK;
;  621          0682  2
;  622          0683  2     LOCAL
;  623          0684  2         ADDR,
;  624          0685  2         STATUS;
;  625          0686  2
;  626          0687  2     ! Note that we store the address of the context area in a temporary.
;  627          0688  2     ! This creates a "window of vulnerability" during which an asynchronous
;  628          0689  2     ! abort could seriously disrupt everything.
;  629          0690  2     !
;  630          0691  2     STATUS = LIB$GET_VM(%REF(CTX_K_SIZE*%UPVAL), ADDR);
;  631          0692  2     IF NOT .STATUS
;  632          0693  2     THEN
;  633          0694  2         RETURN SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS);
;  634          0695  2
;  635          0696  2     ! Store the address of the context area in our global register
;  636          0697  2     !
;  637          0698  2     CTX = .ADDR;
;  638          0699  2
;  639          0700  2     ! Verify that the allocated storage is aligned on a longword boundary
;  640          0701  2     !
;  641          0702  2     IF (CTX[BASE_] AND %B'11') NEQ 0
;  642          0703  2     THEN
```

```
;  643        0704  2          RETURN SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS);
;  644        0705  2
;  645        0706  2      ! Zero-fill the storage before storing it's address
;  646        0707  2      !
;  647        0708  2      CH$FILL(0, CTX_K_SIZE*%UPVAL, CTX[BASE_]);
;  648        0709  2
;  649        0710  2      ! Store the address of the context area, leaving the sync bit alone.
;  650        0711  2      !
;  651        0712  2      CTXADR[0] = .CTXADR[0] OR .CTX;
;  652        0713  2
;  653        0714  2      ! Store the address of the context longword,
;  654        0715  2      ! So we can pass it to the callback routines.
;  655        0716  2      !
;  656        0717  2  %IF NOT HOSTILE %THEN
;  657        0718  2      IF CTXADR[0] NEQ SOR_CONTEXT
;  658        0719  2      THEN
;  659        0720  2  %FI
;  660        0721  2          CTX[COM_CTXADR] = CTXADR[0];
;  661        0722  2
;  662        0723  1      END;


                                                  53  DD 00000 INI_CTX:PUSHL   R3                                  ; 0647
                                 5E               0C  C2 00002         SUBL2   #12, SP                             ;
                          04     AE               53  D0 00005         MOVL    R3, CTXADR                          ;
                                            08     AE  9F 00009         PUSHAB  ADDR                                ; 0691
                          04     AE        0150   8F  3C 0000C         MOVZWL  #336, 4(SP)                         ;
                                            04     AE  9F 00012         PUSHAB  4(SP)                               ;
                   00000000G     00               02  FB 00015         CALLS   #2, LIB$GET_VM                      ;
                                 52               50  D0 0001C         MOVL    R0, STATUS                          ;
                                 09               52  E9 0001F         BLBC    STATUS, 1$                          ; 0692
                                 5B        08     AE  D0 00022         MOVL    ADDR, CTX                           ; 0698
                                 03               5B  93 00026         BITB    CTX, #3                             ; 0702
                                                  10  13 00029         BEQL    2$                                  ;
                                                  52  DD 0002B 1$:     PUSHL   STATUS                              ; 0704
                                                  7E  D4 0002D         CLRL    -(SP)                               ;
                                      001C11B4    8F  DD 0002F         PUSHL   #1839540                            ;
                          B4     AF               03  FB 00035         CALLS   #3, SOR$$ERROR                      ;
                                                  1E  11 00039         BRB     3$                                  ;
   0150   8F          00             6E           00  2C 0003B 2$:     MOVC5   #0, (SP), #0, #336, (CTX)           ; 0708
                                                  6B     00042                                                     ;
                          04     BE               5B  C8 00043         BISL2   CTX, @CTXADR                        ; 0712
                                 50 00000000'     EF  9E 00047         MOVAB   SOR_CONTEXT, R0                     ; 0718
                                 50        04     AE  D1 0004E         CMPL    CTXADR, R0                          ;
                                                  05  13 00052         BEQL    3$                                  ;
                          54     AB        04     AE  D0 00054         MOVL    CTXADR, 84(CTX)                     ; 0721
                                 5E               0C  C0 00059 3$:     ADDL2   #12, SP                             ; 0723
                                                  08  BA 0005C         POPR    #^M<R3>                             ;
                                                      05 0005E         RSB                                         ;

; Routine Size:  95 bytes,    Routine Base:  SOR$RO_CODE + 011A
```

```
1 GLOBAL ROUTINE SOR$$ALLOCATE
1         (
1         SIZE
1         ):      CAL_CTXREG =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine allocates storage.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       SIZE.rl.v       Number of bytes to allocate
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       The process region may be expanded.
1 !
1 ! ROUTINE VALUE:
1 !
1 !       Address of the storage.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 ! NOTES:
1 !
1 !       The storage allocated by this routine is cleared.
1 !       This routine should be used only for small pieces of memory.
1 !       For larger pieces of memory, LIB$GET_VM should be called directly.
1 !
1 !--
2     BEGIN
2     EXTERNAL REGISTER
2         CTX =   COM_REG_CTX:    REF CTX_BLOCK;
2     LOCAL
2         ADDR,
2         STATUS;
2     STATUS = LIB$GET_VM(SIZE, ADDR);
2     IF NOT .STATUS THEN SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS);
2     CH$FILL(0, .SIZE, .ADDR);                    ! Zero-fill the storage
2     RETURN .ADDR;                                ! Return the address
1     END;


                          003C 00000        .ENTRY  SOR$$ALLOCATE, Save R2,R3,R4,R5          ; 0724
          5E            04  C2 00002        SUBL2   #4, SP                                   ;
                        5E  DD 00005        PUSHL   SP                                       ; 0768
```

```
                                    04   AC  9F 00007        PUSHAB  SIZE                                  ;
                  00000000G  00          02  FB 0000A        CALLS   #2, LIB$GET_VM                        ;
                             0F          50  E8 00011        BLBS    STATUS, 1$                            ; 0769
                                         50  DD 00014        PUSHL   STATUS                                ;
                                         7E  D4 00016        CLRL    -(SP)                                 ;
                          001C11B4       8F  DD 00018        PUSHL   #1839540                              ;
                     FF6B    CF          03  FB 0001E        CALLS   #3, SOR$$ERROR                        ;
   04   AC          00      6E          00  2C 00023 1$:    MOVC5   #0, (SP), #0, SIZE, @ADDR             ; 0770
                                    00   BE     00029                                                      ;
                             50          6E  D0 0002B        MOVL    ADDR, R0                              ; 0771
                                             04 0002E        RET                                           ; 0772

; Routine Size:  47 bytes,    Routine Base:  SOR$RO_CODE + 0179
```

```
;  714     0773  1 GLOBAL ROUTINE SOR$$DEALLOCATE
;  715     0774  1         (
;  716     0775  1         SIZE,
;  717     0776  1         ADDR:   REF VECTOR[1],
;  718     0777  1         NEWVAL
;  719     0778  1         ):      CAL_CTXREG NOVALUE =
;  720     0779  1
;  721     0780  1 !++
;  722     0781  1 !
;  723     0782  1 ! FUNCTIONAL DESCRIPTION:
;  724     0783  1 !
;  725     0784  1 !       This routine deallocates storage.
;  726     0785  1 !
;  727     0786  1 ! FORMAL PARAMETERS:
;  728     0787  1 !
;  729     0788  1 !       SIZE.rl.v       Number of bytes to deallocate (by value)
;  730     0789  1 !       ADDR.ra.r       Address of storage to deallocate (by reference)
;  731     0790  1 !       NEWVAL.rl.v     New value to store at ADDR.
;  732     0791  1 !
;  733     0792  1 ! IMPLICIT INPUTS:
;  734     0793  1 !
;  735     0794  1 !       NONE
;  736     0795  1 !
;  737     0796  1 ! IMPLICIT OUTPUTS:
;  738     0797  1 !
;  739     0798  1 !       NONE
;  740     0799  1 !
;  741     0800  1 ! ROUTINE VALUE:
;  742     0801  1 !
;  743     0802  1 !       NONE
;  744     0803  1 !
;  745     0804  1 ! SIDE EFFECTS:
;  746     0805  1 !
;  747     0806  1 !       NONE
;  748     0807  1 !
;  749     0808  1 !--
;  750     0809  2     BEGIN
;  751     0810  2     EXTERNAL REGISTER
;  752     0811  2         CTX =   COM_REG_CTX:    REF CTX_BLOCK;
;  753     0812  2     BUILTIN
;  754     0813  2         NULLPARAMETER;
;  755     0814  2     LOCAL
;  756     0815  2         VAL,
;  757     0816  2         STATUS;
;  758     0817  2     IF .SIZE EQL 0 OR .ADDR[0] EQL 0 THEN RETURN;
;  759     0818  2     VAL = 0;
;  760     0819  2     IF NOT NULLPARAMETER(3) THEN VAL = .NEWVAL;
;  761     0820  2     STATUS = LIB$FREE_VM(SIZE, ADDR[0]);
;  762     0821  2     ADDR[0] = .VAL;
;  763     0822  2     IF NOT .STATUS THEN SOR$$ERROR(
;  764     0823  2         SOR$_SHR_SYSERROR AND NOT STS$M_SEVERITY OR STS$K_ERROR, 0, .STATUS);
;  765     0824  1     END;
```

```
                                    0004 00000          .ENTRY  SOR$$DEALLOCATE, Save R2                ; 0773
                                04  AC  D5 00002        TSTL    SIZE                                    ; 0817
                                    38  13 00005        BEQL    2$
                                08  BC  D5 00007        TSTL    @ADDR
                                    33  13 0000A        BEQL    2$
                                    52  D4 0000C        CLRL    VAL                                     ; 0818
                            03          6C  91 0000E    CMPB    (AP), #3                                ; 0819
                                        09  1F 00011    BLSSU   1$
                                0C      AC  D5 00013    TSTL    12(AP)
                                        04  13 00016    BEQL    1$
                            52          0C  AC D0 00018 MOVL    NEWVAL, VAL
                                        08  AC DD 0001C 1$:     PUSHL   ADDR                            ; 0820
                                        04  AC 9F 0001F PUSHAB  SIZE
                    00000000G   00          02  FB 00022    CALLS   #2, LIB$FREE_VM
                            08  BC          52  D0 00029    MOVL    VAL, @ADDR                          ; 0821
                            0F              50  E8 0002D    BLBS    STATUS, 2$                          ; 0822
                                            50  DD 00030    PUSHL   STATUS                              ; 0823
                                            7E  D4 00032    CLRL    -(SP)                               ; 0822
                                001C11B2    8F  DD 00034    PUSHL   #1839538                            ; 0823
                                FF20    CF      03  FB 0003A    CALLS   #3, SOR$$ERROR
                                                04  0003F 2$:     RET                                     ; 0824

; Routine Size:  64 bytes,    Routine Base:  SOR$RO_CODE + 01A8
```

```
;  767          0825  1 %IF NOT HOSTILE %THEN
;  768          0826  1 ROUTINE CLOSE_FILE
;  769          0827  1         (
;  770          0828  1         FAB:    REF BLOCK[,BYTE],
;  771          0829  1         DDB:    REF DDB_BLOCK,
;  772          0830  1         ERROR
;  773          0831  1         ):      CAL_CTXREG NOVALUE =
;  774          0832  1
;  775          0833  1 !++
;  776          0834  1 !
;  777          0835  1 ! FUNCTIONAL DESCRIPTION:
;  778          0836  1 !
;  779          0837  1 !       This routine closes a file.
;  780          0838  1 !
;  781          0839  1 ! FORMAL PARAMETERS:
;  782          0840  1 !
;  783          0841  1 !       FAB     Pointer to the FAB
;  784          0842  1 !       DDB     Pointer to the DDB
;  785          0843  1 !       ERROR   Message to issue if the close fails.
;  786          0844  1 !
;  787          0845  1 ! IMPLICIT INPUTS:
;  788          0846  1 !
;  789          0847  1 !       NONE
;  790          0848  1 !
;  791          0849  1 ! IMPLICIT OUTPUTS:
;  792          0850  1 !
;  793          0851  1 !       NONE
;  794          0852  1 !
;  795          0853  1 ! ROUTINE VALUE:
;  796          0854  1 !
;  797          0855  1 !       NONE
;  798          0856  1 !
;  799          0857  1 ! SIDE EFFECTS:
;  800          0858  1 !
;  801          0859  1 !       NONE
;  802          0860  1 !
;  803          0861  1 !--
;  804          0862  2     BEGIN
;  805          0863  2
;  806          0864  2     ! Copy the IFI (to reference the file),
;  807          0865  2     ! and the FOP (for user-specified options)
;  808          0866  2     !
;  809          0867  2     ! Zero the IFI in the DDB, so we know that this file is closed
;  810          0868  2     !
;  811          0869  2     FAB[FAB$W_IFI] = .DDB[DDB_IFI];
;  812          0870  2     FAB[FAB$L_FOP] = .DDB[DDB_FOP];
;  813          0871  3     IF NOT $CLOSE(FAB=FAB[BASE_])
;  814          0872  2     THEN
;  815          0873  2         SOR$$ERROR(.ERROR, 1, DDB[DDB_NAME],
;  816          0874  2             .FAB[FAB$L_STS], .FAB[FAB$L_STV]);
;  817          0875  2     DDB[DDB_IFI] = 0;
;  818          0876  2
;  819          0877  1     END;


                                                        .EXTRN  SYS$CLOSE
```

```
                                    000C 00000 CLOSE_FILE:
                                                        .WORD   Save R2,R3                      ; 0826
                 52      04   AC   7D 00002             MOVQ    FAB, R2                         ; 0869
           02    A2      0C   A3   B0 00006             MOVW    12(R3), 2(R2)                   ;
           04    A2      10   A3   D0 0000B             MOVL    16(R3), 4(R2)                   ; 0870
                                 52   DD 00010          PUSHL   R2                              ; 0871
     00000000G   00            01   FB 00012             CALLS   #1, SYS$CLOSE                   ;
                 11            50   E8 00019             BLBS    R0, 1$                          ;
                 7E      08   A2   7D 0001C             MOVQ    8(R2), -(SP)                    ; 0874
                         04   A3   9F 00020             PUSHAB  4(R3)                           ; 0873
                                 01   DD 00023          PUSHL   #1                              ;
                         0C   AC   DD 00025             PUSHL   ERROR                           ;
         FEF2    CF            05   FB 00028             CALLS   #5, SOR$$ERROR                  ;
                         0C   A3   D4 0002D 1$:         CLRL    12(R3)                          ; 0875
                                    04 00030             RET                                     ; 0877

; Routine Size:  49 bytes,    Routine Base:  SOR$RO_CODE + 01E8

;  820          0878  1 %FI
```

```
;  822     0879  1 %IF NOT HOSTILE %THEN
;  823     0880  1 GLOBAL ROUTINE SOR$PASS_FILES                   ! Finished
;  824     0881  1         (
;  825     0882  1         INP_DESC:       REF BLOCK[,BYTE],       ! Input file name descriptor
;  826     0883  1         OUT_DESC:       REF BLOCK[,BYTE],       ! Output file name descriptor
;  827     0884  1         ORG:            REF VECTOR[1,BYTE],     ! File organization
;  828     0885  1         RFM:            REF VECTOR[1,BYTE],     ! Record format
;  829     0886  1         BKS:            REF VECTOR[1,BYTE],     ! Bucket size
;  830     0887  1         BLS:            REF VECTOR[1,WORD],     ! Block size
;  831     0888  1         MRS:            REF VECTOR[1,WORD],     ! Maximum record size
;  832     0889  1         ALQ:            REF VECTOR[1,LONG],     ! Allocation quantity
;  833     0890  1         FOP:            REF VECTOR[1,LONG],     ! File-processing options
;  834     0891  1         FSZ:            REF VECTOR[1,BYTE],     ! VFC field size
;  835     0892  1         CONTEXT:        REF VECTOR[1,LONG]      ! Addr of context longword
;  836     0893  1         ;OUT_CTX
;  837     0894  1         ): AND_RETURN_R1 =
;  838     0895  1 !++
;  839     0896  1 !
;  840     0897  1 ! FUNCTIONAL DESCRIPTION:
;  841     0898  1 !
;  842     0899  1 !       This routine passes input file name(s), output file name, and output
;  843     0900  1 !       file characteristics to the sort or merge.
;  844     0901  1 !
;  845     0902  1 ! FORMAL PARAMETERS:
;  846     0903  1 !
;  847     0904  1 !       INP_DESC.ra.d   Input file descriptor
;  848     0905  1 !       OUT_DESC.ra.d   Output file descriptor
;  849     0906  1 !       ORG.rbu.r       File organization       (FAB$B_ORG)
;  850     0907  1 !       RFM.rbu.r       Record format           (FAB$B_RFM)
;  851     0908  1 !       BKS.rbu.r       Bucket size             (FAB$B_BKS)
;  852     0909  1 !       BLS.rwu.r       Block size              (FAB$W_BLS)
;  853     0910  1 !       MRS.rwu.r       Longest record length   (FAB$W_MRS)
;  854     0911  1 !       ALQ.rlu.r       Allocation quantity     (FAB$L_ALQ)
;  855     0912  1 !       FOP.rlu.r       File options            (FAB$L_FOP)
;  856     0913  1 !       FSZ.rbu.r       VFC field size          (FAB$B_FSZ)
;  857     0914  1 !       CONTEXT.ra.r    Longword pointing to work area
;  858     0915  1 !
;  859     0916  1 !       All parameters except INP_DESC are optional.
;  860     0917  1 !
;  861     0918  1 ! IMPLICIT INPUTS:
;  862     0919  1 !
;  863     0920  1 !       NONE
;  864     0921  1 !
;  865     0922  1 ! IMPLICIT OUTPUTS:
;  866     0923  1 !
;  867     0924  1 !       NONE
;  868     0925  1 !
;  869     0926  1 ! ROUTINE VALUE:
;  870     0927  1 !
;  871     0928  1 !       Status code.
;  872     0929  1 !
;  873     0930  1 ! SIDE EFFECTS:
;  874     0931  1 !
;  875     0932  1 !       If present, the output file characteristics are saved.
;  876     0933  1 !
;  877     0934  1 !--
;  878     0935  2     BEGIN
```

```
;  879          0936  2     FIRSTPARAMETER_(INP_DESC);  ! Required by PRESENT_ and NULL_ macros
;  880          0937  2
;  881          0938  2     LOCAL
;  882          0939  2         DDB:    REF DDB_BLOCK,          ! Pointer to FAB/RAB/NAM/XAB blocks
;  883          0940  2         STATUS;                         ! Status
;  884          0941  2     BUILTIN
;  885          0942  2         ACTUALCOUNT;
;  886          0943  2
;  887          0944  2     CONTEXT_(CONTEXT);
;  888          0945  2
;  889          0946  2     IF ACTUALCOUNT() NEQ 0 THEN
;  890          0947  2     IF .CTX[COM_FLO_NOINIT] THEN RETURN__(SOR$_SORT_ON);
;  891          0948  2
;  892          0949  2     !+
;  893          0950  2     !
;  894          0951  2     ! Process the input file specification
;  895          0952  2     !
;  896          0953  2     !-
;  897          0954  2     IF NOT NULL_(INP_DESC)
;  898          0955  2     THEN
;  899          0956  3         BEGIN
;  900          0957  3         IF .CTX[COM_NUM_FILES] GEQ MAX_FILES THEN RETURN__(SOR$_INP_FILES);
;  901          0958  3
;  902          0959  3         ! Allocate DDB and link into list (at the end of the list)
;  903          0960  3         !
;  904          0961  3         DDB = SOR$$ALLOCATE(DDB_K_SIZE);
;  905          0962  3         DDB[DDB_FIL] = .CTX[COM_NUM_FILES];             ! Numbers 0 on up
;  906          0963  4         BEGIN
;  907          0964  4         LOCAL
;  908          0965  4             P:  REF DDB_BLOCK;
;  909          0966  4         P = CTX[COM_INP_DDB] - %FIELDEXPAND(DDB_NEXT,0);
;  910          0967  4         DECR I FROM .CTX[COM_NUM_FILES]-1 TO 0 DO P = .P[DDB_NEXT];
;  911          0968  4         P[DDB_NEXT] = DDB[BASE_];                       ! Link DDB into list
;  912          0969  4         CTX[COM_NUM_FILES] = .CTX[COM_NUM_FILES] + 1;   ! One more file
;  913          0970  3         END;
;  914          0971  3
;  915          0972  3
;  916          0973  3         ! Copy the input file name string to the DDB
;  917          0974  3         !
;  918          0975  3         SOR$$COPY_FILE_NAME(INP_DESC[BASE_], DDB[DDB_NAME]);
;  919          0976  2         END;
;  920          0977  2
;  921          0978  2
;  922          0979  2     !+
;  923          0980  2     !
;  924          0981  2     ! Process the output file specification
;  925          0982  2     !
;  926          0983  2     !-
;  927          0984  2     IF NOT NULL_(OUT_DESC)
;  928          0985  2     THEN
;  929          0986  3         BEGIN
;  930          0987  3         IF .CTX[COM_OUT_DDB] NEQ 0              ! Output already specified?
;  931          0988  3         THEN
;  932          0989  3             RETURN__(SOR$_DUP_OUTPUT);
;  933          0990  3
;  934          0991  3
;  935          0992  3         ! Allocate the DDB for the output file.
```

```
;  936          0993  3          ! Allocate space to save the ORG, RFM, et.al.
;  937          0994  3          ! This is used in INIT_SORT or INIT_MERGE.
;  938          0995  3          !
;  939          0996  3          CTX[COM_OUT_DDB] = SOR$$ALLOCATE(DDB_K_SIZE);
;  940          0997  3          DDB = .CTX[COM_OUT_DDB];
;  941          0998  4          BEGIN
;  942          0999  4          MACRO
;  943        M 1000  4              FETCH_(N,XXX) =
;  944          1001  4                  IF NOT NULL_(XXX) THEN (P[0]=.P[0] OR 1^N; P[N] = .XXX[0]) %;
;  945          1002  4          LOCAL
;  946          1003  4              P:  REF VECTOR;
;  947          1004  4          P = CTX[COM_PASS_FILES] = SOR$$ALLOCATE(%UPVAL+8*%UPVAL);
;  948          1005  4          FETCH_(1,ORG);
;  949          1006  4          FETCH_(2,RFM);
;  950          1007  4          FETCH_(3,BKS);
;  951          1008  4          FETCH_(4,BLS);
;  952          1009  4          FETCH_(5,MRS);
;  953          1010  4          FETCH_(6,ALQ);
;  954          1011  4          FETCH_(7,FOP);
;  955          1012  4          FETCH_(8,FSZ);
;  956          1013  3          END;
;  957          1014  3
;  958          1015  3
;  959          1016  3          ! Copy the output file name string to the DDB
;  960          1017  3          !
;  961          1018  3          SOR$$COPY_FILE_NAME(OUT_DESC[BASE_], DDB[DDB_NAME]);
;  962          1019  3
;  963          1020  2          END;
;  964          1021  2
;  965          1022  2      OUT_CTX = CTX[BASE_];
;  966          1023  2
;  967          1024  2      RETURN__(SS$_NORMAL);
;  968          1025  1      END;


                                    08FC 00000          .ENTRY  SOR$PASS_FILES, Save R2,R3,R4,R5,R6,R7,R11  ; 0880
                 57 00000000G   00   9E 00002          MOVAB   SOR$$COPY_FILE_NAME, R7                     ;
                 56     FF53    CF   9E 00009          MOVAB   SOR$$ALLOCATE, R6                           ;
                                7E   D4 0000E          CLRL    CTX__                                       ; 0935
                 6D     0170    CF   DE 00010          MOVAL   23$, (FP)                                   ;
                 0B             6C   91 00015          CMPB    (AP), #11                                   ; 0944
                                06   1F 00018          BLSSU   1$                                          ;
                 53       2C    AC   D0 0001A          MOVL    CONTEXT, P__                                ;
                                07   12 0001E          BNEQ    2$                                          ;
                 53 00000000'   EF   9E 00020  1$:      MOVAB   SOR_CONTEXT, P__                            ;
                 6E             53   D0 00027  2$:      MOVL    P__, CTX__                                  ;
           1F    63             00   E2 0002A          BBSS    #0, (P__), 5$                               ;
           5B    63             01   CB 0002E          BICL3   #1, (P__), CTX                              ;
                                03   12 00032          BNEQ    3$                                          ;
                       A1       A6   16 00034          JSB     INI_CTX                                     ;
           0D       5C    AB   04   E2 00037  3$:      BBSS    #4, 92(CTX), 4$                             ;
                 0130    CB   01   D0 0003C          MOVL    #1, 304(CTX)                                ;
                              6C   95 00041          TSTB    (AP)                                        ; 0946
                              58   13 00043          BEQL    10$                                         ;
```

```
        0C        5C   AB  E9 00045          BLBC    92(CTX), 6$                      ; 0947
   00   BE             01  8A 00049 4$:      BICB2   #1, @CTX                         ;
        50 001C802C    8F  D0 0004D 5$:      MOVL    #1867820, R0                     ;
                           04 00054          RET                                      ;
                       46  13 00055 6$:      BEQL    10$                              ; 0954
                  04   AC  D5 00057          TSTL    4(AP)                            ;
                       41  13 0005A          BEQL    10$                              ;
        0A        59   AB  91 0005C          CMPB    89(CTX), #10                     ; 0957
                       0C  1F 00060          BLSSU   7$                               ;
   00   BE             01  8A 00062          BICB2   #1, @CTX                         ;
        50 001C80CC    8F  D0 00066          MOVL    #1867980, R0                     ;
                           04 0006D          RET                                      ;
        7E        59   8F  9A 0006E 7$:      MOVZBL  #89, -(SP)                       ; 0961
        66             01  FB 00072          CALLS   #1, SOR$$ALLOCATE                ;
        52             50  D0 00075          MOVL    R0, DDB                          ;
   58   A2        59   AB  90 00078          MOVB    89(CTX), 88(DDB)                 ; 0962
        51      009C   CB  9E 0007D          MOVAB   156(R11), P                      ; 0966
        50        59   AB  9A 00082          MOVZBL  89(CTX), I                       ; 0967
                       03  11 00086          BRB     9$                               ;
        51             61  D0 00088 8$:      MOVL    (P), P                           ;
        FA             50  F4 0008B 9$:      SOBGEQ  I, 8$                            ;
        61             52  D0 0008E          MOVL    DDB, (P)                         ; 0968
                  59   AB  96 00091          INCB    89(CTX)                          ; 0969
                  04   A2  9F 00094          PUSHAB  4(DDB)                           ; 0975
                  04   AC  DD 00097          PUSHL   INP_DESC                         ;
        67             02  FB 0009A          CALLS   #2, SOR$$COPY_FILE_NAME          ;
        02             6C  91 0009D 10$:     CMPB    (AP), #2                         ; 0984
                       03  1E 000A0          BGEQU   12$                              ;
                     00CE  31 000A2 11$:     BRW     22$                              ;
                  08   AC  D5 000A5 12$:     TSTL    8(AP)                            ;
                       F8  13 000A8          BEQL    11$                              ;
                0098   CB  D5 000AA          TSTL    152(CTX)                         ; 0987
                       0C  13 000AE          BEQL    13$                              ;
   00   BE             01  8A 000B0          BICB2   #1, @CTX                         ; 0989
        50 001C80DC    8F  D0 000B4          MOVL    #1867996, R0                     ;
                           04 000BB          RET                                      ;
        7E        59   8F  9A 000BC 13$:     MOVZBL  #89, -(SP)                       ; 0996
        66             01  FB 000C0          CALLS   #1, SOR$$ALLOCATE                ;
 0098   CB             50  D0 000C3          MOVL    R0, 152(CTX)                     ;
        52      0098   CB  D0 000C8          MOVL    152(CTX), DDB                    ; 0997
                       24  DD 000CD          PUSHL   #36                              ; 1004
        66             01  FB 000CF          CALLS   #1, SOR$$ALLOCATE                ;
 0094   CB             50  D0 000D2          MOVL    R0, 148(CTX)                     ;
        03             6C  91 000D7          CMPB    (AP), #3                         ; 1005
                       0D  1F 000DA          BLSSU   14$                              ;
                  0C   AC  D5 000DC          TSTL    12(AP)                           ;
                       08  13 000DF          BEQL    14$                              ;
        60             02  88 000E1          BISB2   #2, (P)                          ;
   04   A0        0C   BC  9A 000E4          MOVZBL  @ORG, 4(P)                       ;
        04             6C  91 000E9 14$:     CMPB    (AP), #4                         ; 1006
                       0D  1F 000EC          BLSSU   15$                              ;
                  10   AC  D5 000EE          TSTL    16(AP)                           ;
                       08  13 000F1          BEQL    15$                              ;
        60             04  88 000F3          BISB2   #4, (P)                          ;
   08   A0        10   BC  9A 000F6          MOVZBL  @RFM, 8(P)                       ;
        05             6C  91 000FB 15$:     CMPB    (AP), #5                         ; 1007
                       0D  1F 000FE          BLSSU   16$                              ;
```

```
                        14   AC  D5 00100          TSTL    20(AP)                          ;
                             08  13 00103          BEQL    16$                             ;
                60           08  88 00105          BISB2   #8, (P)                         ;
        0C      A0      14   BC  9A 00108          MOVZBL  @BKS, 12(P)                     ;
                06           6C  91 0010D 16$:     CMPB    (AP), #6                        ; 1008
                             0D  1F 00110          BLSSU   17$                             ;
                        18   AC  D5 00112          TSTL    24(AP)                          ;
                             08  13 00115          BEQL    17$                             ;
                60           10  88 00117          BISB2   #16, (P)                        ;
        10      A0      18   BC  3C 0011A          MOVZWL  @BLS, 16(P)                     ;
                07           6C  91 0011F 17$:     CMPB    (AP), #7                        ; 1009
                             0D  1F 00122          BLSSU   18$                             ;
                        1C   AC  D5 00124          TSTL    28(AP)                          ;
                             08  13 00127          BEQL    18$                             ;
                60           20  88 00129          BISB2   #32, (P)                        ;
        14      A0      1C   BC  3C 0012C          MOVZWL  @MRS, 20(P)                     ;
                08           6C  91 00131 18$:     CMPB    (AP), #8                        ; 1010
                             0E  1F 00134          BLSSU   19$                             ;
                        20   AC  D5 00136          TSTL    32(AP)                          ;
                             09  13 00139          BEQL    19$                             ;
                60      40   8F  88 0013B          BISB2   #64, (P)                        ;
        18      A0      20   BC  D0 0013F          MOVL    @ALQ, 24(P)                     ;
                09           6C  91 00144 19$:     CMPB    (AP), #9                        ; 1011
                             0E  1F 00147          BLSSU   20$                             ;
                        24   AC  D5 00149          TSTL    36(AP)                          ;
                             09  13 0014C          BEQL    20$                             ;
                60      80   8F  88 0014E          BISB2   #128, (P)                       ;
        1C      A0      24   BC  D0 00152          MOVL    @FOP, 28(P)                     ;
                0A           6C  91 00157 20$:     CMPB    (AP), #10                       ; 1012
                             0E  1F 0015A          BLSSU   21$                             ;
                        28   AC  D5 0015C          TSTL    40(AP)                          ;
                             09  13 0015F          BEQL    21$                             ;
        01      A0           01  88 00161          BISB2   #1, 1(P)                        ;
        20      A0      28   BC  9A 00165          MOVZBL  @FSZ, 32(P)                     ;
                        04   A2  9F 0016A 21$:     PUSHAB  4(DDB)                          ; 1018
                        08   AC  DD 0016D          PUSHL   OUT_DESC                        ;
                67           02  FB 00170          CALLS   #2, SOR$$COPY_FILE_NAME         ;
                51           5B  D0 00173 22$:     MOVL    CTX, OUT_CTX                    ; 1022
        00      BE           01  8A 00176          BICB2   #1, @CTX                        ; 1024
        5C      AB           10  8A 0017A          BICB2   #16, 92(CTX)                    ;
                50    0130   CB  D0 0017E          MOVL    304(CTX), R0                    ;
                                 04 00183          RET                                     ; 1025
                               0000 00184 23$:     .WORD   Save nothing                    ; 0935
                50      08   AC  D0 00186          MOVL    8(AP), R0                       ;
                50      04   A0  D0 0018A          MOVL    4(R0), R0                       ;
                        FC   A0  9F 0018E          PUSHAB  CTX__                           ;
                             01  DD 00191          PUSHL   #1                              ;
                             5E  DD 00193          PUSHL   SP                              ;
                7E      04   AC  7D 00195          MOVQ    4(AP), -(SP)                    ;
        FC67    CF           03  FB 00199          CALLS   #3, COND_HAND                   ;
                                 04 0019E          RET                                     ;

; Routine Size:  415 bytes,    Routine Base:  SOR$RO_CODE + 0219

;  969          1026  1 %FI
```

```
;  971          1027  1 !+
;  972          1028  1 !
;  973          1029  1 ! Due to the similarity of SOR$BEGIN_SORT and SOR$BEGIN_MERGE,
;  974          1030  1 ! a macro is used to generate code for each.
;  975          1031  1 !
;  976          1032  1 !-
;  977          1033  1 MACRO
;  978        M 1034  1     BEGIN_SORT_MERGE(SM) =
;  979        M 1035  1     FIRSTPARAMETER_(KEY_BUFFER);        ! Required by PRESENT_ and NULL_ macros
;  980        M 1036  1
;  981        M 1037  1     LITERAL
;  982        M 1038  1         USED_OPTIONS =
;  983        M 1039  1             MASK_(
;  984        M 1040  1                 OPT_NODUPS, OPT_NOSIGNAL, OPT_LOAD_FILL, OPT_STABLE
;  985        M 1041  1                 %IF NOT HOSTILE %THEN , OPT_EBCDIC, OPT_MULTI %FI
;  986        M 1042  1                 %IF %IDENTICAL(SM,MERGE) %THEN , OPT_SEQ_CHECK %FI
;  987        M 1043  1                 );
;  988        M 1044  1
;  989        M 1045  1     BUILTIN
;  990        M 1046  1         TESTBITSS,
;  991        M 1047  1         TESTBITSC,
;  992        M 1048  1         ACTUALCOUNT;
;  993        M 1049  1
;  994        M 1050  1     LOCAL
;  995        M 1051  1         DOKEY_RTN,
;  996        M 1052  1         DOKEY_PRM,
;  997        M 1053  1         STATUS;
;  998        M 1054  1
;  999        M 1055  1     ! Get the context area
; 1000        M 1056  1     !
; 1001        M 1057  1     CONTEXT_(CONTEXT);
; 1002        M 1058  1
; 1003        M 1059  1
; 1004        M 1060  1     ! Determine whether to signal or return errors
; 1005        M 1061  1     !
; 1006        M 1062  1     IF (IF NULL_(OPTIONS) THEN TRUE ELSE NOT .OPTIONS[OPT_NOSIGNAL])
; 1007        M 1063  1     THEN
; 1008        M 1064  1         CTX[COM_SIGNAL] = TRUE;
; 1009        M 1065  1
; 1010        M 1066  1
; 1011        M 1067  1     ! Check and update the flow-of-control flags
; 1012        M 1068  1     !
; 1013        M 1069  1     IF TESTBITSS(CTX[COM_FLO_NOINIT]) THEN RETURN__(SOR$_SORT_ON);
; 1014        M 1070  1     %IF %IDENTICAL(SM,SORT)
; 1015        M 1071  1     %THEN
; 1016        M 1072  1         ! If a sort with record interface on input,
; 1017        M 1073  1         ! RELEASE_REC may now be called.
; 1018        M 1074  1         ! Also, SORT_MERGE may be called.
; 1019        M 1075  1         !
; 1020        M 1076  1         IF .CTX[COM_NUM_FILES] EQL 0 THEN CTX[COM_FLO_RELEASE] = TRUE;
; 1021        M 1077  1         CTX[COM_FLO_SORT] = TRUE;
; 1022        M 1078  1     %FI
; 1023        M 1079  1     %IF %IDENTICAL(SM,MERGE)
; 1024        M 1080  1     %THEN
; 1025        M 1081  1         ! If a merge with record interface on output,
; 1026        M 1082  1         ! RETURN_REC may now be called.
; 1027        M 1083  1         ! Also, DO_MERGE (an archaic routine) may be called.
```

```
;  1028        M 1084  1          !
;  1029        M 1085  1          IF .CTX[COM_OUT_DDB] EQL 0 THEN CTX[COM_FLO_RETURN] = TRUE;
;  1030        M 1086  1          CTX[COM_FLO_DOMERGE] = TRUE;
;  1031        M 1087  1      %FI
;  1032        M 1088  1
;  1033        M 1089  1
;  1034        M 1090  1      ! Get the options parameters
;  1035        M 1091  1      !
;  1036        M 1092  1      IF NOT NULL_(OPTIONS)
;  1037        M 1093  1      THEN
;  1038        M 1094  1          BEGIN
;  1039        M 1095  1          IF (.OPTIONS[0,L_] AND NOT USED_OPTIONS) NEQ 0
;  1040        M 1096  1          THEN
;  1041        M 1097  1              RETURN_(SOR$_UNDOPTION);          ! Invalid options specified
;  1042        M 1098  1          %IF %IDENTICAL(SM,MERGE)
;  1043        M 1099  1          %THEN
;  1044        M 1100  1              IF .OPTIONS[OPT_SEQ_CHECK] THEN CTX[COM_SEQ_CHECK] = TRUE;
;  1045        M 1101  1          %FI
;  1046        M 1102  1          IF .OPTIONS[OPT_STABLE] THEN CTX[COM_STABLE] = TRUE;
;  1047        M 1103  1          IF .OPTIONS[OPT_NODUPS] THEN CTX[COM_NODUPS] = TRUE;
;  1048        M 1104  1          IF .OPTIONS[OPT_LOAD_FILL] THEN CTX[COM_LOAD_FILL] = TRUE;
;  1049        M 1105  1
;  1050        M 1106  1          %IF NOT HOSTILE %THEN
;  1051        M 1107  1          IF .OPTIONS[OPT_EBCDIC] OR .OPTIONS[OPT_MULTI]
;  1052        M 1108  1          THEN
;  1053        M 1109  1              BEGIN
;  1054        M 1110  1              !
;  1055        M 1111  1              ! Simply note that we have a collating sequence, so that we won't
;  1056        M 1112  1              ! process other collating sequences in the specification text.
;  1057        M 1113  1              ! The collating sequence will be fully processed after we have an
;  1058        M 1114  1              ! opportunity to process the pad characater, if any.
;  1059        M 1115  1              !
;  1060        M 1116  1              IF .OPTIONS[OPT_EBCDIC] AND .OPTIONS[OPT_MULTI]
;  1061        M 1117  1              THEN
;  1062        M 1118  1                  RETURN_(SOR$_KEYAMBINC);
;  1063        M 1119  1              CTX[COM_OVR_COLSEQ] = TRUE;
;  1064        M 1120  1              END;
;  1065        M 1121  1          %FI
;  1066        M 1122  1          END;
;  1067        M 1123  1
;  1068        M 1124  1      %IF %IDENTICAL(SM,SORT)
;  1069        M 1125  1      %THEN
;  1070        M 1126  1
;  1071        M 1127  1          ! Determine the type of sort
;  1072        M 1128  1          !
;  1073        M 1129  1          IF NOT NULL_(SORT_TYPE)
;  1074        M 1130  1          THEN
;  1075        M 1131  1              BEGIN
;  1076        M 1132  1              CTX[COM_SORT_TYPE] = .SORT_TYPE[0];
;  1077        M 1133  1              CTX[COM_OVR_PROC] = TRUE;
;  1078        M 1134  1              END;
;  1079        M 1135  1          IF .CTX[COM_SORT_TYPE] EQL 0
;  1080        M 1136  1          THEN
;  1081        M 1137  1              CTX[COM_SORT_TYPE] = TYP_K_RECORD   ! Default to a record sort
;  1082        M 1138  1          ELIF
;  1083        M 1139  1              .CTX[COM_SORT_TYPE] GTRU TYP_K_MAX  ! Check for valid sort type
;  1084        M 1140  1          THEN
```

```
;  1085        M 1141  1              RETURN__( SOR$$ERROR(SOR$_BAD_TYPE) );
;  1086        M 1142  1
;  1087        M 1143  1
;  1088        M 1144  1          ! Get the total file allocation, if specified.
;  1089        M 1145  1          !
;  1090        M 1146  1          IF NOT NULL_(FILE_ALLOC) THEN CTX[COM_FILE_ALLOC] = .FILE_ALLOC[0];
;  1091        M 1147  1
;  1092        M 1148  1
;  1093        M 1149  1          ! Get the number of work files, and check the value.
;  1094        M 1150  1          ! Note: we don't open the work files 'til needed.
;  1095        M 1151  1          !
;  1096        M 1152  1          CTX[COM_WRK_FILES] = DEF_WORK_FILES;
;  1097        M 1153  1          IF NOT NULL_(WORK_FILES)
;  1098        M 1154  1          THEN
;  1099        M 1155  1              BEGIN
;  1100        M 1156  1              CTX[COM_WRK_FILES] = .WORK_FILES[0];
;  1101        M 1157  1              IF .CTX[COM_WRK_FILES] NEQ 0
;  1102        M 1158  1              THEN
;  1103        M 1159  1                  IF .CTX[COM_WRK_FILES] LSSU MIN_WORK_FILES
;  1104        M 1160  1                  THEN
;  1105        M 1161  1                      BEGIN
;  1106        M 1162  1                      CTX[COM_WRK_FILES] = MIN_WORK_FILES;
;  1107        M 1163  1                      SOR$$ERROR(SOR$_BAD_MERGE);
;  1108        M 1164  1                      END
;  1109        M 1165  1                  ELIF .CTX[COM_WRK_FILES] GTRU MAX_WORK_FILES
;  1110        M 1166  1                  THEN
;  1111        M 1167  1                      BEGIN
;  1112        M 1168  1                      CTX[COM_WRK_FILES] = MAX_WORK_FILES;
;  1113        M 1169  1                      SOR$$ERROR(SOR$_BAD_MERGE);
;  1114        M 1170  1                      END;
;  1115        M 1171  1              END;
;  1116        M 1172  1      %FI
;  1117        M 1173  1
;  1118        M 1174  1
;  1119        M 1175  1      %IF %IDENTICAL(SM,MERGE)
;  1120        M 1176  1      %THEN
;  1121        M 1177  1          CTX[COM_MERGE] = TRUE;
;  1122        M 1178  1
;  1123        M 1179  1
;  1124        M 1180  1          ! Set the type of sort to record so that SOR$$KEY_SUB doesn't screw up.
;  1125        M 1181  1          !
;  1126        M 1182  1          CTX[COM_SORT_TYPE] = TYP_K_RECORD;
;  1127        M 1183  1
;  1128        M 1184  1
;  1129        M 1185  1          ! Get the merge order.
;  1130        M 1186  1          !
;  1131        M 1187  1          CTX[COM_MRG_ORDER] = .CTX[COM_NUM_FILES];
;  1132        M 1188  1          IF .CTX[COM_MRG_ORDER] EQL 0
;  1133        M 1189  1          THEN
;  1134        M 1190  1              BEGIN
;  1135        M 1191  1              IF NOT NULL_(MERGE_ORDER)
;  1136        M 1192  1              THEN
;  1137        M 1193  1                  CTX[COM_MRG_ORDER] = .MERGE_ORDER[0];
;  1138        M 1194  1              IF .CTX[COM_MRG_ORDER] EQL 0 OR
;  1139        M 1195  1                 .CTX[COM_MRG_ORDER] GTRU MAX_MERGE_ORDER
;  1140        M 1196  1              THEN
;  1141        M 1197  1                  RETURN__( SOR$$ERROR(SOR$_BAD_MERGE) );
```

```
;  1142          M 1198  1                END;
;  1143          M 1199  1        %FI
;  1144          M 1200  1
;  1145          M 1201  1
;  1146          M 1202  1        ! Get the LRL value if specified.
;  1147          M 1203  1        !
;  1148          M 1204  1        IF NOT NULL_(LRL)
;  1149          M 1205  1        THEN
;  1150          M 1206  1            BEGIN
;  1151          M 1207  1            CTX[COM_LRL] = .LRL[0];
;  1152          M 1208  1            %IF 1^%FIELDEXPAND(COM_LRL,2)-1 GTRU MAX_REFSIZE %THEN
;  1153          M 1209  1            IF .CTX[COM_LRL] GTRU MAX_REFSIZE THEN RETURN__(SOR$_BAD_LRL);
;  1154          M 1210  1            %FI
;  1155          M 1211  1            END;
;  1156          M 1212  1
;  1157          M 1213  1
;  1158          M 1214  1        ! Save the address of the equal-key routine
;  1159          M 1215  1        !
;  1160          M 1216  1        IF NOT NULL_(USER_EQUAL)
;  1161          M 1217  1        THEN
;  1162          M 1218  1            BEGIN
;  1163          M 1219  1            CTX[COM_EQUAL] = .USER_EQUAL;
;  1164          M 1220  1            IF TESTBITSC(CTX[COM_NODUPS]) THEN SOR$$ERROR(SOR$_NODUPEXC);
;  1165          M 1221  1            IF TESTBITSC(CTX[COM_STABLE]) THEN SOR$$ERROR(SOR$_STABLEEXC);
;  1166          M 1222  1            END
;  1167          M 1223  1        ELIF .CTX[COM_NODUPS]
;  1168          M 1224  1        THEN
;  1169          M 1225  1            IF TESTBITSC(CTX[COM_STABLE]) THEN SOR$$ERROR(SOR$_STABLEEXC);
;  1170          M 1226  1
;  1171          M 1227  1
;  1172          M 1228  1        ! Indicate whether the keys in the specification text should be ignored
;  1173          M 1229  1        !
;  1174          M 1230  1        IF NOT NULL_(USER_COMPARE) OR NOT NULL_(KEY_BUFFER)
;  1175          M 1231  1        THEN
;  1176          M 1232  1            CTX[COM_OVR_KEY] = TRUE;
;  1177          M 1233  1
;  1178          M 1234  1
;  1179          M 1235  1        ! Process the specification file or specification text
;  1180          M 1236  1        !
;  1181          M 1237  1        %IF NOT HOSTILE
;  1182          M 1238  1        %THEN
;  1183          M 1239  1            IF .CTX[COM_SPC_DDB] NEQ 0 OR
;  1184          M 1240  1                .BLOCK[CTX[COM_SPC_TXT], DSC$W_LENGTH; ,BYTE] NEQ 0
;  1185          M 1241  1            THEN
;  1186          M 1242  1                SOR$$SPEC_FILE();
;  1187          M 1243  1        %FI
;  1188          M 1244  1
;  1189          M 1245  1        %IF NOT HOSTILE %THEN
;  1190          M 1246  1        IF NOT NULL_(OPTIONS) THEN
;  1191          M 1247  1        IF .OPTIONS[OPT_EBCDIC] OR .OPTIONS[OPT_MULTI] OR
;  1192          M 1248  1            .CTX[COM_PTAB] NEQ 0
;  1193          M 1249  1        THEN
;  1194          M 1250  1            BEGIN
;  1195          M 1251  1            !
;  1196          M 1252  1            ! Now that we know the pad character, process EBCDIC/DEC_MULTI.
;  1197          M 1253  1            !
;  1198          M 1254  1            LOCAL
```

```
;  1199          M 1255  1              PAD_CHAR:   VECTOR[4,BYTE],
;  1200          M 1256  1              COL_DESC:   VECTOR[2];
;  1201          M 1257  1
;  1202          M 1258  1          IF .CTX[COM_WRK_SIZ] EQL 0
;  1203          M 1259  1          THEN
;  1204          M 1260  1              BEGIN
;  1205          M 1261  1              CTX[COM_WRK_SIZ] = WRK_K_COLLATE;
;  1206          M 1262  1              STATUS = LIB$GET_VM(CTX[COM_WRK_SIZ], CTX[COM_WRK_ADR]);
;  1207          M 1263  1              IF NOT .STATUS THEN SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS);
;  1208          M 1264  1              CTX[COM_WRK_END] = .CTX[COM_WRK_ADR] + .CTX[COM_WRK_SIZ];
;  1209          M 1265  1              END;
;  1210          M 1266  1
;  1211          M 1267  1          COLL_DESC[0] = .CTX[COM_WRK_END] - .CTX[COM_WRK_ADR];
;  1212          M 1268  1          COLL_DESC[1] = .CTX[COM_WRK_ADR];
;  1213          M 1269  1          STATUS = COLL$INIT(COLL_DESC[0]);
;  1214          M 1270  1          IF NOT .STATUS THEN RETURN_(.STATUS);
;  1215          M 1271  1          IF    .OPTIONS[OPT_MULTI]  THEN STATUS = SOR$$DECM  (COLL_DESC[0])
;  1216          M 1272  1          ELIF .OPTIONS[OPT_EBCDIC] THEN STATUS = SOR$$EBCDIC(COLL_DESC[0])
;  1217          M 1273  1          ELSE STATUS = COL$BASE(COLL_DESC[0], .CTX[COM_PTAB]);
;  1218          M 1274  1          IF NOT .STATUS THEN RETURN__(.STATUS);
;  1219          M 1275  1          PAD_CHAR[0] = 1;
;  1220          M 1276  1          PAD_CHAR[1] = 0;
;  1221          M 1277  1          PAD_CHAR[2] = .CTX[COM_PAD];
;  1222          M 1278  1          STATUS = COLL$PAD(COLL_DESC[0], PAD_CHAR[0]);
;  1223          M 1279  1          IF NOT .STATUS THEN RETURN__(.STATUS);
;  1224          M 1280  1          IF .CTX[COM_TIE_BREAK]
;  1225          M 1281  1          THEN
;  1226          M 1282  1              BEGIN
;  1227          M 1283  1              STATUS = COLL$TIE_BREAK(COLL_DESC[0], TRUE);
;  1228          M 1284  1              IF NOT .STATUS THEN RETURN__(.STATUS);
;  1229          M 1285  1              END;
;  1230          M 1286  1          STATUS = COLL$RESULT(COLL_DESC[0], COLL_DESC[0]);
;  1231          M 1287  1          IF NOT .STATUS THEN RETURN_(.STATUS);
;  1232          M 1288  1          CTX[COM_WRK_ADR] = .CTX[COM_WRK_ADR] + .COLL_DESC[0];
;  1233          M 1289  1          CTX[COM_COLATE] = .COLL_DESC[1];
;  1234          M 1290  1          END;
;  1235          M 1291  1      %FI
;  1236          M 1292  1
;  1237          M 1293  1
;  1238          M 1294  1      ! If record interface on input, assume variable-length records,
;  1239          M 1295  1      ! unless (see below) the OPT_FIXED option was requested.
;  1240          M 1296  1      ! Also, check the LRL value, and sort process.
;  1241          M 1297  1      !
;  1242          M 1298  1      IF .CTX[COM_NUM_FILES] EQL 0
;  1243          M 1299  1      THEN
;  1244          M 1300  1          BEGIN
;  1245          M 1301  1          CTX[COM_VAR] = TRUE;              ! Records are variable-length
;  1246          M 1302  1          IF .CTX[COM_SORT_TYPE] NEQ TYP_K_RECORD THEN RETURN__(SOR$_BAD_TYPE);
;  1247          M 1303  1          IF .CTX[COM_LRL] EQL 0 THEN RETURN__(SOR$_LRL_MISS);    ! LRL required
;  1248          M 1304  1          END;
;  1249          M 1305  1
;  1250          M 1306  1
;  1251          M 1307  1      !+
;  1252          M 1308  1      !
;  1253          M 1309  1      ! There is a data-flow problem with the fields COM_LRL (the longest input
;  1254          M 1310  1      ! record length), COM_LRL_OUT (the longest output record length), and
;  1255          M 1311  1      ! COM_MINVFC (the size of the VFC area that must be saved in the internal
```

```
;  1256        M 1312  1     ! format nodes for record sorts).
;  1257        M 1313  1     !
;  1258        M 1314  1     ! COM_LRL is not known until after the input files are opened.
;  1259        M 1315  1     ! COM_LRL_OUT is calculated by key processing (and depends on COM_LRL).
;  1260        M 1316  1     ! COM_LRL_OUT is used in opening the output file.
;  1261        M 1317  1     ! COM_MINVFC is calculated after opening the output file.
;  1262        M 1318  1     ! COM_MINVFC is used by key processing.
;  1263        M 1319  1     !
;  1264        M 1320  1     ! Rather than split SOR$$OPEN and SOR$$KEY_SUB into two pieces each, and
;  1265        M 1321  1     ! call the pieces in the appropriate order, SOR$$KEY_SUB is called from the
;  1266        M 1322  1     ! middle of SOR$$OPEN.
;  1267        M 1323  1     !
;  1268        M 1324  1     ! The requirements on the routines are as follows:
;  1269        M 1325  1     !   Before calling SOR$$KEY_SUB, SOR$$OPEN must set COM_LRL and must set
;  1270        M 1326  1     !       COM_MINVFC as large as may be needed.
;  1271        M 1327  1     !   SOR$$KEY_SUB must set COM_LRL_OUT.
;  1272        M 1328  1     !   Either:
;  1273        M 1329  1     !       After calling SOR$$KEY_SUB, SOR$$OPEN may decrease COM_MINVFC, but
;  1274        M 1330  1     !       if it was non-zero for SOR$$KEY_SUB, the VFC area must still be
;  1275        M 1331  1     !       allocated (otherwise, the input routine will access violate).
;  1276        M 1332  1     !   Or:
;  1277        M 1333  1     !       SOR$$KEY_SUB, when saving the VFC area, must first check whether
;  1278        M 1334  1     !       the storage is, indeed, allocated.
;  1279        M 1335  1     !
;  1280        M 1336  1     ! The effects of this method are as follows:
;  1281        M 1337  1     !   The high-water mark on the amount of stack space used is higher.
;  1282        M 1338  1     !   Some performance will be lost if the internal format records retain the
;  1283        M 1339  1     !       VFC fields even when they are not really needed.  This occurs if
;  1284        M 1340  1     !       the input files have VFC format, and the output file is overlaid,
;  1285        M 1341  1     !       and the output file does not have VFC format.  This should be rare.
;  1286        M 1342  1     !-
;  1287        M 1343  1
;  1288        M 1344  1
;  1289        M 1345  1     ! The user must have specified the keys by either:
;  1290        M 1346  1     ! 1. Passing a user key comparison routine.
;  1291        M 1347  1     ! 2. Passing the key buffer information, or
;  1292        M 1348  1     ! 3. Using the specification file to specify the keys.
;  1293        M 1349  1     !
;  1294        M 1350  1     ! These are used in the above order.
;  1295        M 1351  1     !
;  1296        M 1352  1     ! We want to process the keys sometime after opening the input files,
;  1297        M 1353  1     ! but before opening the input files.  Set up a routine and a parameter
;  1298        M 1354  1     ! that SOR$$OPEN will call at the appropriate time.  This will:
;  1299        M 1355  1     ! Build the comparison routine, and the input and output conversion
;  1300        M 1356  1     ! routines, if needed.
;  1301        M 1357  1     !
;  1302        M 1358  1     DOKEY_RTN = SOR$$KEY_SUB;
;  1303        M 1359  1     DOKEY_PRM = 0;
;  1304        M 1360  1     IF NOT NULL_(USER_COMPARE)
;  1305        M 1361  1     THEN
;  1306        M 1362  1         BEGIN
;  1307        M 1363  1         CTX[COM_COMPARE] = .USER_COMPARE;
;  1308        M 1364  1         IF NOT NULL_(KEY_BUFFER) THEN SOR$$ERROR(SOR$_KEYAMBINC);
;  1309        M 1365  1         END
;  1310        M 1366  1     ELIF NOT NULL_(KEY_BUFFER)
;  1311        M 1367  1     THEN
;  1312        M 1368  1         BEGIN
```

```
;  1313        M 1369  1          DOKEY_PRM = KEY_BUFFER[0];
;  1314        M 1370  1          END
;  1315        M 1371  1      %IF NOT HOSTILE %THEN
;  1316        M 1372  1      ELIF
;  1317        M 1373  1          .CTX[COM_RDT_SIZ] NEQ 0
;  1318        M 1374  1      THEN
;  1319        M 1375  1          BEGIN
;  1320        M 1376  1          !
;  1321        M 1377  1          ! Use the keys as specified in the specification file
;  1322        M 1378  1          ! Note that the parameter to SOR$$SPEC_KEY_SUB will be ignored.
;  1323        M 1379  1          !
;  1324        M 1380  1          DOKEY_RTN = SOR$$SPEC_KEY_SUB;
;  1325        M 1381  1          END
;  1326        M 1382  1      %FI
;  1327        M 1383  1      ELSE
;  1328        M 1384  1          BEGIN
;  1329        M 1385  1          !
;  1330        M 1386  1          ! Use the default key specification (SOR$$KEY_SUB can figure it out)
;  1331        M 1387  1          !
;  1332        M 1388  1          0;
;  1333        M 1389  1          END;
;  1334        M 1390  1
;  1335        M 1391  1
;  1336        M 1392  1      ! Open the input files and output file now.
;  1337        M 1393  1      ! Since the longest output record length is needed before the output file
;  1338        M 1394  1      ! is created, pass a routine and parameter that will calculate it.
;  1339        M 1395  1      !
;  1340        M 1396  1      ! The parameters to SOR$$OPEN are a routine and parameter that gets called
;  1341        M 1397  1      ! after opening the input files, but before opening the output file.
;  1342        M 1398  1      !
;  1343        M 1399  1      %IF HOSTILE
;  1344        M 1400  1      %THEN
;  1345        M 1401  1          IF .CTX[COM_FILE_ALLOC] EQL 0 THEN CTX[COM_FILE_ALLOC] = DEF_FILE_ALLOC;
;  1346        M 1402  1          STATUS = CAL_CTXREG(.DOKEY_RTN, .DOKEY_PRM );
;  1347        M 1403  1      %ELSE
;  1348        M 1404  1          STATUS = SOR$$OPEN( .DOKEY_RTN, .DOKEY_PRM );
;  1349        M 1405  1      %FI
;  1350        M 1406  1      IF NOT .STATUS THEN RETURN  (.STATUS);
;  1351        M 1407  1      SOR$$DEALLOCATE(%UPVAL+8*%OPVAL, CTX[COM_PASS_FILES]);
;  1352        M 1408  1
;  1353        M 1409  1
;  1354        M 1410  1      ! Allocate space for the record header buffer (VFC area)
;  1355        M 1411  1      ! This is done here, rather than in SORT_MERGE, since SORT_MERGE need
;  1356        M 1412  1      ! not be called for merges.
;  1357        M 1413  1      !
;  1358        M 1414  1      IF .CTX[COM_MAXVFC] NEQ 0
;  1359        M 1415  1      THEN
;  1360        M 1416  1          BEGIN
;  1361        M 1417  1          CTX[COM_RHB_INP] = SOR$$ALLOCATE(.CTX[COM_MAXVFC]);
;  1362        M 1418  1          CTX[COM_RHB_OUT] = SOR$$ALLOCATE(.CTX[COM_MAXVFC]);
;  1363        M 1419  1          END;
;  1364        M 1420  1
;  1365        M 1421  1
;  1366        M 1422  1      ! If the user specified fixed-length records, make them fixed-length
;  1367        M 1423  1      ! The effect of this is that no length will be stored in the internal
;  1368        M 1424  1      ! format nodes.
;  1369        M 1425  1      !
```

```
;  1370        M 1426  1     %IF (USED_OPTIONS AND MASK_(OPT_FIXED)) NEQ 0 %THEN
;  1371        M 1427  1     %ERROR('This should be done before calling SOR$$KEY_SUB')
;  1372        M 1428  1     IF NOT NULL_(OPTIONS) THEN
;  1373        M 1429  1     IF .OPTIONS[OPT_FIXED] THEN
;  1374        M 1430  1     IF .CTX[COM_VAR]
;  1375        M 1431  1     THEN
;  1376        M 1432  1         BEGIN
;  1377        M 1433  1         !
;  1378        M 1434  1         ! If COM_VAR was set by SOR$$OPEN (variable length files), complain.
;  1379        M 1435  1         ! If COM_VAR was set because the record interface is being used on
;  1380        M 1436  1         ! input, just clear COM_VAR.
;  1381        M 1437  1         !
;  1382        M 1438  1         IF .CTX[COM_NUM_FILES] NEQ 0 THEN SOR$$ERROR(SOR$_VAR_FIX);
;  1383        M 1439  1         CTX[COM_VAR] = FALSE;
;  1384        M 1440  1         CTX[COM_SRL] = .CTX[COM_LRL];   ! Performance only -- affects tree size
;  1385        M 1441  1         END;
;  1386        M 1442  1     %FI
;  1387        M 1443  1
;  1388        M 1444  1
;  1389        M 1445  1     ! For a non-record sort, create an array containing addresses of the DDBs.
;  1390        M 1446  1     !
;  1391        M 1447  1     IF .CTX[COM_SORT_TYPE] NEQ TYP_K_RECORD
;  1392        M 1448  1     THEN
;  1393        M 1449  1         BEGIN
;  1394        M 1450  1         LOCAL
;  1395        M 1451  1             P:  REF DDB_BLOCK,          ! Pointer to DDB
;  1396        M 1452  1             V:  REF VECTOR;             ! Pointer to array
;  1397        M 1453  1         CTX[COM_INP_ARRAY] = V = SOR$$ALLOCATE(%UPVAL * .CTX[COM_NUM_FILES]);
;  1398        M 1454  1         P = .CTX[COM_INP_DDB];
;  1399        M 1455  1         DECR I FROM .CTX[COM_NUM_FILES]-1 TO 0 DO
;  1400        M 1456  1             BEGIN
;  1401        M 1457  1             V[.P[DDB_FIL]] = P[BASE_];
;  1402        M 1458  1             P = .P[DDB_NEXT];
;  1403        M 1459  1             END;
;  1404        M 1460  1         END;
;  1405        M 1461  1
;  1406          1462  1     %;
```

```
;  1408          1463  1 GLOBAL ROUTINE SOR$BEGIN_SORT
;  1409          1464  1         (
;  1410          1465  1         KEY_BUFFER:     REF VECTOR[1,WORD],     ! Key description buffer
;  1411          1466  1         LRL:            REF VECTOR[1,WORD],     ! Longest record length
;  1412          1467  1         OPTIONS:        REF BLOCK[1],           ! Options
;  1413          1468  1         FILE_ALLOC:     REF VECTOR[1,LONG],     ! Input file(s) allocation
;  1414          1469  1         USER_COMPARE,                           ! User comparison routine
;  1415          1470  1         USER_EQUAL,                             ! User equal-key routine
;  1416          1471  1         SORT_TYPE:      REF VECTOR[1,BYTE],     ! Type of sort process
;  1417          1472  1         WORK_FILES:     REF VECTOR[1,BYTE],     ! Number of work files
;  1418          1473  1         CONTEXT:        REF VECTOR[1,LONG]      ! Context longword
;  1419          1474  1         ) =
;  1420          1475  1 !++
;  1421          1476  1 !
;  1422          1477  1 ! FUNCTIONAL DESCRIPTION:
;  1423          1478  1 !
;  1424          1479  1 !       This routine initializes sort.
;  1425          1480  1 !       Input and output files are opened.
;  1426          1481  1 !       Key definitions are processed.
;  1427          1482  1 !       The replacement selection tree is initialized.
;  1428          1483  1 !
;  1429          1484  1 ! FORMAL PARAMETERS:
;  1430          1485  1 !
;  1431          1486  1 !       KEY_BUFFER.raw          Key buffer address
;  1432          1487  1 !       LRL.rwu.r               Longest record length
;  1433          1488  1 !       FILE_ALLOC.rlu.r        Input file allocation
;  1434          1489  1 !       WORK_FILES.rbu.r        Number of work files
;  1435          1490  1 !       SORT_TYPE.rbu.r         Type of sort (record/tag/index/address)
;  1436          1491  1 !       USER_COMPARE.rzem.r     User-written comparison routine
;  1437          1492  1 !       OPTIONS.rlu.r           Option bits
;  1438          1493  1 !       USER_EQUAL.rzem.r       User-written equal-key routine
;  1439          1494  1 !       CONTEXT.ra.r            Longword pointing to work area
;  1440          1495  1 !
;  1441          1496  1 !       All parameters are optional.
;  1442          1497  1 !
;  1443          1498  1 ! IMPLICIT INPUTS:
;  1444          1499  1 !
;  1445          1500  1 !       NONE
;  1446          1501  1 !
;  1447          1502  1 ! IMPLICIT OUTPUTS:
;  1448          1503  1 !
;  1449          1504  1 !       NONE
;  1450          1505  1 !
;  1451          1506  1 ! ROUTINE VALUE:
;  1452          1507  1 !
;  1453          1508  1 !       Status code.
;  1454          1509  1 !
;  1455          1510  1 ! SIDE EFFECTS:
;  1456          1511  1 !
;  1457          1512  1 !       The working set is extended and the virtual memory is allocated.
;  1458          1513  1 !
;  1459          1514  1 !--
;  1460          1515  2     BEGIN
;  1461          1516  2
;  1462          1517  2     ! Process the parameters
;  1463          1518  2     !
;  1464          1519  2     BEGIN_SORT_MERGE(SORT)
```

```
; 1465          1520  2
; 1466          1521  2
; 1467          1522  2      ! Determine amount of memory available for the replacement selection tree,
; 1468          1523  2      ! based on the working-set extent, and initialize the tree.
; 1469          1524  2      !
; 1470          1525  3      BEGIN
; 1471          1526  3      LOCAL
; 1472          1527  3          WSSIZE,
; 1473          1528  3          WSLIMIT,
; 1474          1529  3          MEM;
; 1475          1530  3 !    STATUS = $ADJWSL(PAGCNT=0, WSETLM=WSSIZE);
; 1476          1531  3 !    IF NOT .STATUS THEN RETURN__(SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS));
; 1477        P 1532  3      STATUS = $ADJWSL(PAGCNT=%X'7FFFFFFF', WSETLM=WSLIMIT
; 1478          1533  3          %IF HOSTILE_ELAN %THEN , ELAN = .CTX[COM_CTXADR] %FI );
; 1479          1534  3      IF NOT .STATUS THEN RETURN__(SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS));
; 1480        L 1535  3      %IF TUN_K_PURGWS
; 1481        U 1536  3      %THEN
; 1482        U 1537  3          $PURGWS(INADR=UPLIT(0,%X'7FFFFFFF'));
; 1483          1538  3      %FI
; 1484          1539  3      !
; 1485          1540  3      ! Call TREE_INIT, passing the number of pages it may use,
; 1486          1541  3      ! and the expected number of input records.
; 1487          1542  3      !
; 1488          1543  3      ! Note that we effectively add only 1 to the average record length for
; 1489          1544  3      ! variable-length records.  This has the effect of slightly increasing
; 1490          1545  3      ! our estimate of the number of records.
; 1491          1546  3      !
; 1492          1547  3      STATUS = SOR$$TREE_INIT(
; 1493          1548  3          MAX(.WSLIMIT - TUN_K_NONTREE, TUN_K_FALLBACK),
; 1494          1549  3          .CTX[COM_FILE_ALLOC] * COM_K_BPERBLOCK * 2 /
; 1495          1550  4              (IF NOT .CTX[COM_VAR]
; 1496          1551  4                  THEN 2*.CTX[COM_LRL]
; 1497          1552  3                  ELSE .CTX[COM_LRL]+.CTX[COM_SRL] + 2));
; 1498          1553  3      IF NOT .STATUS THEN RETURN__(.STATUS);
; 1499          1554  2      END;
; 1500          1555  2
; 1501          1556  2
; 1502          1557  2      RETURN__(SS$_NORMAL);
; 1503          1558  1      END;


                                                                  .EXTRN  SYS$ADJWSL

                                        087C 00000                .ENTRY  SOR$BEGIN_SORT, Save R2,R3,R4,R5,R6,R11      ; 1463
                  56      FD49    CF   9E 00002                MOVAB   SOR$$ERROR, R6                                ;
                  5E              14   C2 00007                SUBL2   #20, SP                                       ;
                            10    AE   D4 0000A                CLRL    CTX__                                         ; 1515
                  6D      03F8    CF   DE 0000D                MOVAL   59$, (FP)                                     ;
                  09              6C   91 00012                CMPB    (AP), #9                                      ; 1519
                                  06   1F 00015                BLSSU   1$                                            ;
                  53        24    AC   D0 00017                MOVL    CONTEXT, P__                                  ;
                                  07   12 0001B                BNEQ    2$                                            ;
                  53 00000000'    EF   9E 0001D 1$:            MOVAB   SOR_CONTEXT, P__                              ;
            10    AE              53   D0 00024 2$:            MOVL    P__, CTX__                                    ;
      2F          63              00   E2 00028                BBSS    #0, (P__), 7$                                 ;
      5B          63              01   CB 0002C                BICL3   #1, (P__), CTX                                ;
```

```
                                 03  12 00030          BNEQ    3$                                                                      ;
                              13 A6  16 00032          JSB     INI_CTX                                                                 ;
                1D     5C AB    04  E2 00035 3$:       BBSS    #4, 92(CTX), 6$                                                         ;
                     0130 CB    01  D0 0003A          MOVL    #1, 304(CTX)                                                            ;
                          03    6C  91 0003F          CMPB    (AP), #3                                                                ;
                                0A  1F 00042          BLSSU   4$                                                                      ;
                              0C AC  D5 00044          TSTL    12(AP)                                                                  ;
                                05  13 00047          BEQL    4$                                                                      ;
                04    0C BC       03  E0 00049          BBS     #3, @OPTIONS, 5$                                                        ;
                      5B AB       04  88 0004E 4$:       BISB2   #4, 91(CTX)                                                             ;
                0C    5C AB       00  E3 00052 5$:       BBCS    #0, 92(CTX), 8$                                                         ;
                      10 BE       01  8A 00057 6$:       BICB2   #1, @CTX                                                                ;
                         50 001C802C  8F  D0 0005B 7$:       MOVL    #1867820, R0                                                        ;
                                     04 00062          RET                                                                             ;
                         52       58 AB  9E 00063 8$:       MOVAB   88(CTX), R2                                                         ;
                                  01 A2  95 00067          TSTB    1(R2)                                                                   ;
                                     04  12 0006A          BNEQ    9$                                                                      ;
                         5C AB       02  88 0006C          BISB2   #2, 92(CTX)                                                             ;
                         03 A2    80 8F  88 00070 9$:       BISB2   #128, 3(R2)                                                         ;
                            03       6C  91 00075          CMPB    (AP), #3                                                                ;
                                     55  1F 00078          BLSSU   16$                                                                     ;
                                  0C AC  D5 0007A          TSTL    12(AP)                                                                  ;
                                     50  13 0007D          BEQL    16$                                                                     ;
                            50    0C AC  D0 0007F          MOVL    OPTIONS, R0                                                             ;
                    FFFFFDB0    8F       60  D3 00083          BITL    (R0), #-592                                                         ;
                                         0C  13 0008A          BEQL    10$                                                                 ;
                         10 BE           01  8A 0008C          BICB2   #1, @CTX                                                            ;
                            50 001C814C  8F  D0 00090          MOVL    #1868108, R0                                                        ;
                                         04 00097          RET                                                                         ;
                            04           60  E9 00098 10$:       BLBC    (R0), 11$                                                       ;
                         03 A2           01  88 0009B          BISB2   #1, 3(R2)                                                           ;
                04       60              06  E1 0009F 11$:       BBC     #6, (R0), 12$                                                   ;
                         03 A2           20  88 000A3          BISB2   #32, 3(R2)                                                          ;
                C4       60              09  E1 000A7 12$:       BBC     #9, (R0), 13$                                                   ;
                         03 A2           10  88 000AB          BISB2   #16, 3(R2)                                                          ;
                08       60              01  E0 000AF 13$:       BBS     #1, (R0), 14$                                                   ;
                18       60              02  E1 000B3          BBC     #2, (R0), 16$                                                   ;
                10       60              01  E1 000B7          BBC     #1, (R0), 15$                                                   ;
                0C       60              02  E1 000BB 14$:       BBC     #2, (R0), 15$                                                   ;
                         10 BE           01  8A 000BF          BICB2   #1, @CTX                                                            ;
                            50 001C8134  8F  D0 000C3          MOVL    #1868084, R0                                                        ;
                                         04 000CA          RET                                                                         ;
                         79 AB           04  88 000CB 15$:       BISB2   #4, 121(CTX)                                                        ;
                            07           6C  91 000CF 16$:       CMPB    (AP), #7                                                        ;
                                         0D  1F 000D2          BLSSU   17$                                                                 ;
                                      1C AC  D5 000D4          TSTL    28(AP)                                                              ;
                                         08  13 000D7          BEQL    17$                                                                 ;
                            62        1C BC  90 000D9          MOVB    @SORT_TYPE, (R2)                                                    ;
                         79 AB           01  88 000DD          BISB2   #1, 121(CTX)                                                        ;
                                         62  95 000E1 17$:       TSTB    (R2)                                                            ;
                                         05  12 000E3          BNEQ    18$                                                             ;
                            62           01  90 000E5          MOVB    #1, (R2)                                                        ;
                                         13  11 000E8          BRB     19$                                                             ;
                            04           62  91 000EA 18$:       CMPB    (R2), #4                                                        ;
                                         0E  1B 000ED          BLEQU   19$                                                             ;
                         10 BE           01  8A 000EF          BICB2   #1, @CTX                                                            ;
                               001C806C  8F  DD 000F3          PUSHL   #1867884                                                            ;
```

```
                66          01  FB 000F9          CALLS   #1, SOR$$ERROR                  ;
                                04 000FC          RET                                     ;
                04          6C  91 000FD 19$:     CMPB    (AP), #4                        ;
                            0B  1F 00100          BLSSU   20$                             ;
                      10    AC  D5 00102          TSTL    16(AP)                          ;
                            06  13 00105          BEQL    20$                             ;
          00A8  CB    10    BC  D0 00107          MOVL    @FILE_ALLOC, 168(CTX)           ;
            02  A2          02  90 0010D 20$:     MOVB    #2, 2(R2)                       ;
                08          6C  91 00111          CMPB    (AP), #8                        ;
                            1F  1F 00114          BLSSU   21$                             ;
                      20    AC  D5 00116          TSTL    32(AP)                          ;
                            1A  13 00119          BEQL    21$                             ;
            02  A2    20    BC  90 0011B          MOVB    @WORK_FILES, 2(R2)              ;
                            13  13 00120          BEQL    21$                             ;
                0A    02    A2  91 00122          CMPB    2(R2), #10                      ;
                            0D  1B 00126          BLEQU   21$                             ;
            02  A2          0A  90 00128          MOVB    #10, 2(R2)                      ;
                  001C80BA  8F  DD 0012C          PUSHL   #1867962                        ;
                66          01  FB 00132          CALLS   #1, SOR$$ERROR                  ;
                02          6C  91 00135 21$:     CMPB    (AP), #2                        ;
                            0B  1F 00138          BLSSU   22$                             ;
                      08    AC  D5 0013A          TSTL    8(AP)                           ;
                            06  13 0013D          BEQL    22$                             ;
          0084  CB    08    BC  B0 0013F          MOVW    @LRL, 132(CTX)                  ;
                06          6C  91 00145 22$:     CMPB    (AP), #6                        ;
                            19  1F 00148          BLSSU   23$                             ;
                      18    AC  D5 0014A          TSTL    24(AP)                          ;
                            14  13 0014D          BEQL    23$                             ;
            04  AB    18    AC  D0 0014F          MOVL    USER_EQUAL, 4(CTX)              ;
      0F        62          1D  E5 00154          BBCC    #29, (R2), 24$                  ;
                  001C8104  8F  DD 00158          PUSHL   #1868036                        ;
                66          01  FB 0015E          CALLS   #1, SOR$$ERROR                  ;
                            04  11 00161          BRB     24$                             ;
      0D        62          1D  E1 00163 23$:     BBC     #29, (R2), 25$                  ;
      09        62          18  E5 00167 24$:     BBCC    #24, (R2), 25$                  ;
                  001C8154  8F  DD 0016B          PUSHL   #1868116                        ;
                66          01  FB 00171          CALLS   #1, SOR$$ERROR                  ;
                05          6C  91 00174 25$:     CMPB    (AP), #5                        ;
                            05  1F 00177          BLSSU   26$                             ;
                      14    AC  D5 00179          TSTL    20(AP)                          ;
                            09  12 0017C          BNEQ    27$                             ;
                            6C  95 0017E 26$:     TSTB    (AP)                            ;
                            09  13 00180          BEQL    28$                             ;
                      04    AC  D5 00182          TSTL    4(AP)                           ;
                            04  13 00185          BEQL    28$                             ;
            79  AB          02  88 00187 27$:     BISB2   #2, 121(CTX)                    ;
                    00AC    CB  D5 0018B 28$:     TSTL    172(CTX)                        ;
                            06  12 0018F          BNEQ    29$                             ;
                    00F4    CB  B5 00191          TSTW    244(CTX)                        ;
                            07  13 00195          BEQL    30$                             ;
      00000000G 00          00  FB 00197 29$:     CALLS   #0, SOR$$SPEC_FILE              ;
                03          6C  91 0019E 30$:     CMPB    (AP), #3                        ;
                            03  1E 001A1          BGEQU   32$                             ;
                          00F3  31 001A3 31$:     BRW     42$                             ;
                      0C    AC  D5 001A6 32$:     TSTL    12(AP)                          ;
                            F8  13 001A9          BEQL    31$                             ;
      0B    0C  BC          01  E0 001AB          BBS     #1, @OPTIONS, 33$               ;
```

```
                    06        0C   BC                  02  E0  001B0           BBS     #2, @OPTIONS, 33$                                    ;
                                        011C            CB  D5  001B5           TSTL    284(CTX)                                             ;
                                                        E8  13  001B9           BEQL    31$                                                  ;
                                    52      0124        CB  9E  001BB 33$:      MOVAB   292(CTX), R2                                         ;
                                                        62  D5  001C0           TSTL    (R2)                                                 ;
                                                        2D  12  001C2           BNEQ    35$                                                  ;
                                    62      0600        8F  3C  001C4           MOVZWL  #1536, (R2)                                          ;
                                            0128        CB  9F  001C9           PUSHAB  296(CTX)                                             ;
                                                        52  DD  001CD           PUSHL   R2                                                   ;
                        00000000G   00                  02  FB  001CF           CALLS   #2, LIB$GET_VM                                       ;
                                    54                  50  D0  001D6           MOVL    R0, STATUS                                           ;
                                    0D                  54  E8  001D9           BLBS    STATUS, 34$                                          ;
                                                        54  DD  001DC           PUSHL   STATUS                                               ;
                                                        7E  D4  001DE           CLRL    -(SP)                                                ;
                                        001C11B4        8F  DD  001E0           PUSHL   #1839540                                             ;
                                    66                  03  FB  001E6           CALLS   #3, SOR$$ERROR                                       ;
                012C    CB      0128    CB              62  C1  001E9 34$:      ADDL3   (R2), 296(CTX), 300(CTX)                             ;
                  08    AE      012C    CB      0128    CB  C3  001F1 35$:      SUBL3   296(CTX), 300(CTX), COLL_DESC                        ;
                                  0C    AE      0128    CB  D0  001FA           MOVL    296(CTX), COLL_DESC+4                                ;
                                                    08  AE  9F  00200           PUSHAB  COLL_DESC                                            ;
                        00000000G   00                  01  FB  00203           CALLS   #1, COLL$INIT                                        ;
                                    54                  50  D0  0020A           MOVL    R0, STATUS                                           ;
                                    78                  54  E9  0020D           BLBC    STATUS, 40$                                          ;
                    0C        0C   BC                  02  E1  00210           BBC     #2, @OPTIONS, 36$                                    ;
                                                    08  AE  9F  00215           PUSHAB  COLL_DESC                                            ;
                        00000000G   00                  01  FB  00218           CALLS   #1, SOR$$DECM                                        ;
                                                        1F  11  0021F           BRB     38$                                                  ;
                    0C        0C   BC                  01  E1  00221 36$:      BBC     #1, @OPTIONS, 37$                                    ;
                                                    08  AE  9F  00226           PUSHAB  COLL_DESC                                            ;
                        00000000G   00                  01  FB  00229           CALLS   #1, SOR$$EBCDIC                                      ;
                                                        0E  11  00230           BRB     38$                                                  ;
                                            011C        CB  DD  00232 37$:      PUSHL   284(CTX)                                             ;
                                              0C        AE  9F  00236           PUSHAB  COLL_DESC                                            ;
            00000000G   00                  02  FB  00239           CALLS   #2, COLL$BASE                                        ;
                        54                  50  D0  00240 38$:      MOVL    R0, STATUS                                           ;
                        42                  54  E9  00243           BLBC    STATUS, 40$                                          ;
                        6E                  01  B0  00246           MOVW    #1, PAD_CHAR                                         ;
                  02    AE      0101        CB  90  00249           MOVB    257(CTX), PAD_CHAR+2                                 ;
                                            5E  DD  0024F           PUSHL   SP                                                   ;
                                              0C    AE  9F  00251           PUSHAB  COLL_DESC                                            ;
            00000000G   00                  02  FB  00254           CALLS   #2, COLL$PAD                                         ;
                        54                  50  D0  0025B           MOVL    R0, STATUS                                           ;
                        27                  54  E9  0025E           BLBC    STATUS, 40$                                          ;
                        12          0080    CB  E9  00261           BLBC    128(CTX), 39$                                        ;
                                            01  DD  00266           PUSHL   #1                                                   ;
                                              0C    AE  9F  00268           PUSHAB  COLL_DESC                                            ;
            00000000G   00                  02  FB  0026B           CALLS   #2, COLL$TIE_BREAK                                   ;
                        54                  50  D0  00272           MOVL    R0, STATUS                                           ;
                        10                  54  E9  00275           BLBC    STATUS, 40$                                          ;
                                                08  AE  9F  00278 39$:      PUSHAB  COLL_DESC                                            ;
                                              0C    AE  9F  0027B           PUSHAB  COLL_DESC                                            ;
            00000000G   00                  02  FB  0027E           CALLS   #2, COLL$RESULT                                      ;
                        54                  50  D0  00285           MOVL    R0, STATUS                                           ;
                        03                  54  E8  00288 40$:      BLBS    STATUS, 41$                                          ;
                                        0165    31  0028B           BRW     57$                                                  ;
                0128    CB          08      AE  C0  0028E 41$:      ADDL2   COLL_DESC, 296(CTX)                                  ;
                  68    AB          0C      AE  D0  00294           MOVL    COLL_DESC+4, 104(CTX)                                ;
```

```
                        59   AB  95 00299 42$:    TSTB    89(CTX)
                             29  12 0029C         BNEQ    44$
                0080    CB       02  88 0029E         BISB2   #2, 128(CTX)
                        01       58  AB  91 002A3         CMPB    88(CTX), #1
                             0C  13 002A7         BEQL    43$
                  10    BE       01  8A 002A9         BICB2   #1, @CTX
                        50 001C806C  8F  D0 002AD         MOVL    #1867884, R0
                             04  002B4                   RET
                            0084    CB  B5 002B5 43$:    TSTW    132(CTX)
                             0C  12 002B9         BNEQ    44$
                  10    BE       01  8A 002BB         BICB2   #1, @CTX
                        50 001C8074  8F  D0 002BF         MOVL    #1867892, R0
                             04  002C6                   RET
                        53 00000000G 00  9E 002C7 44$:    MOVAB   SOR$$KEY_SUB, DOKEY_RTN
                             52  D4 002CE         CLRL    DOKEY_PRM
                        05           6C  91 002D0         CMPB    (AP), #5
                             1D  1F 002D3         BLSSU   45$
                                14   AC  D5 002D5         TSTL    20(AP)
                             18  13 002D8         BEQL    45$
                        6B       14   AC  D0 002DA         MOVL    USER_COMPARE, (CTX)
                             6C  95 002DE         TSTB    (AP)
                             2C  13 002E0         BEQL    47$
                                04   AC  D5 002E2         TSTL    4(AP)
                             27  13 002E5         BEQL    47$
                          001C8134   8F  DD 002E7         PUSHL   #1868084
                        66           01  FB 002ED         CALLS   #1, SOR$$ERROR
                             1C  11 002F0         BRB     47$
                             6C  95 002F2 45$:    TSTB    (AP)
                             0B  13 002F4         BEQL    46$
                                04   AC  D5 002F6         TSTL    4(AP)
                             06  13 002F9         BEQL    46$
                           52        04   AC  D0 002FB         MOVL    KEY_BUFFER, DOKEY_PRM
                             0D  11 002FF         BRB     47$
                                00FC    CB  95 00301 46$:    TSTB    252(CTX)
                             07  13 00305         BEQL    47$
                           53 00000000G 00  9E 00307         MOVAB   SOR$$SPEC_KEY_SUB, DOKEY_RTN
                             52  DD 0030E 47$:    PUSHL   DOKEY_PRM
                             53  DD 00310         PUSHL   DOKEY_RTN
                    00000000G   00           02  FB 00312         CALLS   #2, SOR$$OPEN
                             54  50  D0 00319         MOVL    R0, STATUS
                           03            54  E8 0031C         BLBS    STATUS, 48$
                                00D1    31  31 0031F         BRW     57$
                                0094    CB  9F 00322 48$:    PUSHAB  148(CTX)
                             24  DD 00326         PUSHL   #36
                        00A1    C6           02  FB 00328         CALLS   #2, SOR$$DEALLOCATE
                             50  0082    CB  9A 0032D         MOVZBL  130(CTX), R0
                             19  13 00332         BEQL    49$
                             50  DD 00334         PUSHL   R0
                          72    A6           01  FB 00336         CALLS   #1, SOR$$ALLOCATE
                        008C    CB           50  D0 0033A         MOVL    R0, 140(CTX)
                             7E  0082    CB  9A 0033F         MOVZBL  130(CTX), -(SP)
                          72    A6           01  FB 00344         CALLS   #1, SOR$$ALLOCATE
                        0090    CB           50  D0 00348         MOVL    R0, 144(CTX)
                             01       58  AB  91 0034D 49$:    CMPB    88(CTX), #1
                             2A  13 00351         BEQL    52$
                             50       59  AB  9A 00353         MOVZBL  89(CTX), R0
                      7E             50       02  78 00357         ASHL    #2, R0, -(SP)
```

```
          72   A6            01  FB 0035B        CALLS   #1, SOR$$ALLOCATE                  ;
        00A4   CB            50  D0 0035F        MOVL    V, 164(CTX)                        ;
              52     009C    CB  D0 00364        MOVL    156(CTX), P                        ;
              53       59    AB  9A 00369        MOVZBL  89(CTX), I                         ;
                            0B  11 0036D        BRB     51$                                ;
              51       58    A2  9A 0036F 50$:   MOVZBL  88(P), R1                          ;
            6041             52  D0 00373        MOVL    P, (V)[R1]                         ;
              52             62  D0 00377        MOVL    (P), P                             ;
              F2             53  F4 0037A 51$:   SOBGEQ  I, 50$                             ;
                       04    AE  9F 0037D 52$:   PUSHAB  WSLIMIT                            ; 1533
                 7FFFFFFF    8F  DD 00380        PUSHL   #2147483647                        ;
   00000000G  00             02  FB 00386        CALLS   #2, SYS$ADJWSL                     ;
              54             50  D0 0038D        MOVL    R0, STATUS                         ;
              12             54  E8 00390        BLBS    STATUS, 53$                        ; 1534
          10  BE             01  8A 00393        BICB2   #1, @CTX__                         ;
                            54  DD 00397        PUSHL   STATUS                             ;
                            7E  D4 00399        CLRL    -(SP)                              ;
                 001C11B4    8F  DD 0039B        PUSHL   #1839540                           ;
              66             03  FB 003A1        CALLS   #3, SOR$$ERROR                     ;
                                04 003A4        RET                                        ;
  52    00A8   CB            0A  78 003A5 53$:   ASHL    #10, 168(CTX), R2                  ; 1549
              50     0084    CB  9E 003AB        MOVAB   132(CTX), R0                       ; 1551
  0B    0080   CB            01  E0 003B0        BBS     #1, 128(CTX), 54$                  ; 1550
              51             60  3C 003B6        MOVZWL  (R0), R1                           ; 1551
              51             02  C4 003B9        MULL2   #2, R1                             ;
              50             51  D0 003BC        MOVL    R1, R0                             ;
                            0D  11 003BF        BRB     55$                                ;
              51             60  3C 003C1 54$:   MOVZWL  (R0), R1                           ; 1552
              50       02    A0  3C 003C4        MOVZWL  2(R0), R0                          ;
              50             51  C0 003C8        ADDL2   R1, R0                             ;
              50             02  C0 003CB        ADDL2   #2, R0                             ;
  7E          52             50  C7 003CE 55$:   DIVL3   R0, R2, -(SP)                      ; 1550
  50      08  AE 000000C0    8F  C3 003D2        SUBL3   #192, WSLIMIT, R0                  ; 1548
                            50  DD 003DB        PUSHL   R0                                 ;
              3F             6E  D1 003DD        CMPL    (SP), #63                          ;
                            04  14 003E0        BGTR    56$                                ;
              6E       40    8F  9A 003E2        MOVZBL  #64, (SP)                          ;
   00000000G  00             02  FB 003E6 56$:   CALLS   #2, SOR$$TREE_INIT                 ;
              54             50  D0 003ED        MOVL    R0, STATUS                         ;
              08             54  E8 003F0        BLBS    STATUS, 58$                        ; 1553
          10  BE             01  8A 003F3 57$:   BICB2   #1, @CTX__                         ;
              50             54  D0 003F7        MOVL    STATUS, R0                         ;
                                04 003FA        RET                                        ;
          10  BE             01  8A 003FB 58$:   BICB2   #1, @CTX__                         ; 1557
          5C  AB             10  8A 003FF        BICB2   #16, 92(CTX)                       ;
              50     0130    CB  D0 00403        MOVL    304(CTX), R0                       ;
                                04 00408        RET                                        ; 1558
                            0000 00409 59$:     .WORD   Save nothing                       ; 1515
              50       08    AC  D0 0040B        MOVL    8(AP), R0                          ;
              50       04    A0  D0 0040F        MOVL    4(R0), R0                          ;
                       FC    A0  9F 00413        PUSHAB  CTX__                              ;
                            01  DD 00416        PUSHL   #1                                 ;
                            5E  DD 00418        PUSHL   SP                                 ;
              7E       04    AC  7D 0041A        MOVQ    4(AP), -(SP)                       ;
        F843  CF             03  FB 0041E        CALLS   #3, COND_HAND                      ;
                                04 00423        RET                                        ;
```

```
; Routine Size:  1060 bytes,    Routine Base:  SOR$RO_CODE + 03B8
```

```
;1505     1559  1 GLOBAL ROUTINE SOR$SORT_MERGE                    ! Finished
;1506     1560  1         (
;1507     1561  1         CONTEXT:        REF VECTOR[1,LONG]       ! Addr of context longword
;1508     1562  1         ) =
;1509     1563  1 !++
;1510     1564  1 !
;1511     1565  1 ! FUNCTIONAL DESCRIPTION:
;1512     1566  1 !
;1513     1567  1 !       This routine sorts the file.
;1514     1568  1 !
;1515     1569  1 !       For file interface, the input files are read and released to the sort.
;1516     1570  1 !       For record interface, the records have already been released.
;1517     1571  1 !       In either case, the replacement selection tree is flushed, and the
;1518     1572  1 !       merge passes on the runs are done until a single merge will produce
;1519     1573  1 !       a single run.  At this point, the following occurs:
;1520     1574  1 !       For file interface, the output records are reformatted, and directed
;1521     1575  1 !       to the output file.  For record interface, we return to the user, who
;1522     1576  1 !       can call RETURN_REC to get the sorted records.
;1523     1577  1 !
;1524     1578  1 ! FORMAL PARAMETERS:
;1525     1579  1 !
;1526     1580  1 !       CONTEXT.ra.r            Longword pointing to work area
;1527     1581  1 !
;1528     1582  1 !       All parameters are optional.
;1529     1583  1 !
;1530     1584  1 ! IMPLICIT INPUTS:
;1531     1585  1 !
;1532     1586  1 !       NONE
;1533     1587  1 !
;1534     1588  1 ! IMPLICIT OUTPUTS:
;1535     1589  1 !
;1536     1590  1 !       The input files and output file are closed.
;1537     1591  1 !
;1538     1592  1 ! ROUTINE VALUE:
;1539     1593  1 !
;1540     1594  1 !       Status code.
;1541     1595  1 !
;1542     1596  1 ! SIDE EFFECTS:
;1543     1597  1 !
;1544     1598  1 !       NONE
;1545     1599  1 !
;1546     1600  1 !--
;1547     1601  2     BEGIN
;1548     1602  2     FIRSTPARAMETER_(CONTEXT);   ! Required by PRESENT_ and NULL_ macros
;1549     1603  2
;1550     1604  2     LOCAL
;1551     1605  2         DDB:    REF DDB_BLOCK,          ! Pointer to DDB
;1552     1606  2         FAB:    $FAB_DECL,              ! FAB (for closing files)
;1553     1607  2         RAB:    REF $RAB_DECL;          ! Pointer to input or output RAB
;1554     1608  2     BUILTIN
;1555     1609  2         TESTBITCC;
;1556     1610  2
;1557     1611  2     CONTEXT_(CONTEXT, SOR$_SORT_ON);
;1558     1612  2
;1559     1613  2     IF TESTBITCC(CTX[COM_FLO_SORT]) THEN RETURN_(SOR$_SORT_ON);
;1560     1614  2     IF .CTX[COM_OUT_DDB] EQL 0 THEN CTX[COM_FLO_RETURN] = TRUE;
;1561     1615  2
```

```
;  1562        1616  2     %IF NOT HOSTILE %THEN
;  1563        1617  2
;  1564        1618  2     ! Initialize FAB (used for closing files)
;  1565        1619  2     !
;  1566        1620  2     CH$FILL(0, FAB$C_BLN, FAB[BASE_]);
;  1567        1621  2     FAB[FAB$B_BLN] = FAB$C_BLN;
;  1568        1622  2     FAB[FAB$B_BID] = FAB$C_BID;
;  1569        1623  2
;  1570        1624  2
;  1571        1625  2     ! Read all the input files
;  1572        1626  2     !
;  1573        1627  2     DDB = .CTX[COM_INP_DDB];                    ! Point to first DDB
;  1574        1628  2     DECR I FROM .CTX[COM_NUM_FILES]-1 TO 0 DO
;  1575        1629  2         BEGIN
;  1576        1630  2
;  1577        1631  2         ! Store address of the current DDB, and get a pointer to the RAB.
;  1578        1632  2         !
;  1579        1633  2         CTX[COM_INP_CURR] = DDB[BASE_];
;  1580        1634  2         RAB = DDB[DDB_RAB+BASE_];
;  1581        1635  2
;  1582        1636  2
;  1583        1637  2         ! Clear the VFC area.
;  1584        1638  2         ! This area is cleared on a per-file basis, so that extra junk from
;  1585        1639  2         ! the previous file is not left around, and copied to the output file.
;  1586        1640  2         ! We must clear it, as the input conversion routine may reference it,
;  1587        1641  2         ! even if RAB$L_RHB doesn't.
;  1588        1642  2         !
;  1589        1643  2         CH$FILL(0, .CTX[COM_MAXVFC], .CTX[COM_RHB_INP]);
;  1590        1644  3
;  1591        1645  3
;  1592        1646  3         ! If we need the VFC area for internal nodes,
;  1593        1647  3         ! then set RAB$L_RHB to point to the VFC area.
;  1594        1648  3         ! Check to see whether the VFC area is solely for the use of the
;  1595        1649  3         ! output file; if so, don't point the RAB to it.
;  1596        1650  3         !
;  1597        1651  3         RAB[RAB$L_RHB] = .CTX[COM_RHB_INP];
;  1598        1652  3         IF .CTX[COM_MINVFC] EQL 0 THEN RAB[RAB$L_RHB] = 0;
;  1599        1653  3
;  1600        1654  3
;  1601        1655  3         ! Get all the records from this file
;  1602        1656  3         !
;  1603        1657  4         BEGIN
;  1604        1658  4         BUILTIN R0;
;  1605        1659  4         WHILE TRUE DO
;  1606        1660  5             BEGIN
;  1607        1661  6             IF  BEGIN
;  1608        1662  6                 IF (R0 = $GET(RAB=RAB[BASE_])) THEN TRUE
;  1609        1663  6                 ELIF .R0 EQL RMS$_RTB
;  1610        1664  7                 THEN (RAB[RAB$W_RSZ] = .RAB[RAB$L_STV]; TRUE)
;  1611        1665  6                 ELSE FALSE
;  1612        1666  6                 END
;  1613        1667  5             THEN
;  1614        1668  6                 BEGIN
;  1615        1669  6                 LOCAL
;  1616        1670  6                     D:  BLOCK[2];               ! Descriptor
;  1617        1671  6                 MACRO
;  1618        1672  6                     LENW = 0,0,16,0 %,
```

```
;  1619     1673  6              LENL = 0,0,32,0 %,
;  1620     1674  6              ADDR = 1,0,32,0 %;
;  1621     1675  6
;  1622     1676  6          CTX[COM_INP_RECNUM] = .CTX[COM_INP_RECNUM] + 1;
;  1623     1677  6          CHECKPOINT_COUNTDOWN();
;  1624     1678  6
;  1625     1679  6          ! Create a descriptor for the input record
;  1626     1680  6          !
;  1627     1681  6          D[LENL] = .RAB[RAB$W_RSZ];
;  1628     1682  6          D[ADDR] = .RAB[RAB$L_RBF];
;  1629     1683  6          IF
;  1630     1684  7              BEGIN
;  1631     1685  7              !
;  1632     1686  7              ! Check the length.
;  1633     1687  7              !
;  1634     1688  7              IF .D[LENW] GTRU .CTX[COM_LRL]
;  1635     1689  7              THEN
;  1636     1690  8                  BEGIN
;  1637     1691  8                  !
;  1638     1692  8                  ! If too long, complain and shorten the record
;  1639     1693  8                  !
;  1640     1694  8                  R0 = SOR$$ERROR(SOR$_BAD_LRL, 1, .D[LENL],
;  1641     1695  8                      SOR$_SHR_READERR, 1, DDB[DDB_NAME]);
;  1642     1696  8                  D[LENW] = .CTX[COM_LRL];
;  1643     1697  8                  TRUE
;  1644     1698  8                  END
;  1645     1699  7              ELSE
;  1646     1700  7                  .D[LENW] GEQU .CTX[COM_SRL]
;  1647     1701  7              END
;  1648     1702  6          THEN
;  1649     1703  7              BEGIN
;  1650     1704  7              !
;  1651     1705  7              ! Release the record to the sort (also converts the record)
;  1652     1706  7              !
;  1653     1707  7              R0 = SOR$$TREE_INSERT(D[BASE_]);
;  1654     1708  7              END
;  1655     1709  6          ELSE
;  1656     1710  7              BEGIN
;  1657     1711  7              !
;  1658     1712  7              ! If too short, complain and ignore the record.
;  1659     1713  7              !
;  1660     1714  7              ! By putting this code after the code to call TREE_INSERT,
;  1661     1715  7              ! a branch byte displacement after TREE_INSERT suffices.
;  1662     1716  7              !
;  1663     1717  7              R0 = SOR$$ERROR(SOR$_BAD_SRL, 1, .D[LENL],
;  1664     1718  7                  SOR$_SHR_READERR, 1, DDB[DDB_NAME]);
;  1665     1719  7              CTX[COM_OMI_RECNUM] = .CTX[COM_OMI_RECNUM] + 1;
;  1666     1720  7              END
;  1667     1721  6          END
;  1668     1722  5      ELIF
;  1669     1723  5          .R0 EQL RMS$_RSA                ! Record Stream Active
;  1670     1724  5      THEN
;  1671     1725  6          R0 = $WAIT(RAB=RAB[BASE_])      ! Wait until not so active
;  1672     1726  5      ELIF
;  1673     1727  5          .R0 EQL RMS$_RLK
;  1674     1728  5      THEN
;  1675     1729  6          BEGIN
```

```
;  1676     1730  6                BUILTIN
;  1677     1731  6                    TESTBITSS;
;  1678     1732  6                IF TESTBITSS(RAB[RAB$V_RRL]) THEN EXITLOOP;
;  1679     1733  6                RO = SOR$$ERROR(
;  1680     1734  6                    SOR$_SHR_READERR AND NOT STS$M_SEVERITY OR STS$K_WARNING,
;  1681     1735  6                    1, DDB[DDB_NAME], .RAB[RAB$L_STS], .RAB[RAB$L_STV],
;  1682     1736  6                    RMS$_OK_RRL);
;  1683     1737  6                END
;  1684     1738  5            ELSE
;  1685     1739  5                EXITLOOP;                       ! Some other error
;  1686     1740  5
;  1687     1741  4            END;
;  1688     1742  4
;  1689     1743  4
;  1690     1744  4        ! Check for the expected status
;  1691     1745  4        !
;  1692     1746  4        IF .RO NEQ RMS$_EOF
;  1693     1747  4        THEN
;  1694     1748  4            RO = SOR$$ERROR(SOR$_SHR_READERR, 1, DDB[DDB_NAME],
;  1695     1749  4                    .RAB[RAB$L_STS], .RAB[RAB$L_STV]);
;  1696     1750  3        END;
;  1697     1751
;  1698     1752
;  1699     1753  3        ! All records have been read from this file.  Close the input file now
;  1700     1754  3        ! (unless we are doing a tag sort, because we will reaccess the file).
;  1701     1755  3        !
;  1702     1756  3        IF .CTX[COM_SORT_TYPE] NEQ TYP_K_TAG
;  1703     1757  3        THEN
;  1704     1758  3            CLOSE_FILE(FAB[BASE_], DDB[BASE_], SOR$_SHR_CLOSEIN)
;  1705     1759  3        ELSE
;  1706     1760  4            BEGIN
;  1707     1761  4            !
;  1708     1762  4            ! We will access this file later by RFA.
;  1709     1763  4            ! Also, the VFC area (if any) will be needed.
;  1710     1764  4            !
;  1711     1765  4            DDB[DDB_RAB+RAB$B_RAC] = RAB$C_RFA;
;  1712     1766  4            DDB[DDB_RAB+RAB$L_RHB] = .CTX[COM_RHB_INP];
;  1713     1767  3            END;
;  1714     1768  3
;  1715     1769  3
;  1716     1770  3        ! Advance to the next file
;  1717     1771  3        !
;  1718     1772  3        DDB = .DDB[DDB_NEXT];
;  1719     1773  2        END;
;  1720     1774  2
;  1721     1775  2    !+
;  1722     1776  2    ! Now we've read all the input files.
;  1723     1777  2    !
;  1724     1778  2    !-
;  1725     1779  2
;  1726     1780  2
;  1727     1781  2        ! Call another routine to get all the records from TREE_EXTRACT, and to
;  1728     1782  2        ! output the records.  If the record interface is being used on output,
;  1729     1783  2        ! this routine will do nothing.
;  1730     1784  2        !
;  1731     1785  2        MRG_OUTPUT();
;  1732     1786  2        %FI
```

```
; 1733          1787  2
; 1734          1788  2     RETURN__(SS$_NORMAL);
; 1735          1789  1     END;


                                                                  .EXTRN  SYS$GET, SYS$WAIT

                                                  0FFC 00000      .ENTRY  SOR$SORT_MERGE, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1559
                                                                          R9,R10,RT1                                   ;
                                  5E        A0   AE  9E 00002      MOVAB   -96(SP), SP                                  ;
                                            0C   AE  D4 00006      CLRL    CTX__                                        ; 1601
                                  6D      01A9   CF  DE 00009      MOVAL   24$, (FP)                                    ;
                                                 6C  95 0000E      TSTB    (AP)                                         ; 1611
                                                 06  13 00010      BEQL    1$                                           ;
                                  53        04   AC  D0 00012      MOVL    CONTEXT, P__                                 ;
                                                 07  12 00016      BNEQ    2$                                           ;
                                  53 00000000'   EF  9E 00018 1$:  MOVAB   SOR_CONTEXT, P__                             ;
                            0C    AE             53  D0 0001F 2$:  MOVL    P__, CTX__                                   ;
                    19            63             00  E2 00023      BBSS    #0, (P__), 4$                                ;
                    5B            63             01  CB 00027      BICL3   #1, (P__), CTX                               ;
                                                 0F  13 0002B      BEQL    3$                                           ;
                    0A      5C    AB             04  E2 0002D      BBSS    #4, 92(CTX), 3$                              ;
                          0130    CB             01  D0 00032      MOVL    #1, 304(CTX)                                 ;
                    0C      58    AB             1F  E4 00037      BBSC    #31, 88(CTX), 5$                             ; 1613
                            0C    BE             01  8A 0003C 3$:  BICB2   #1, @CTX__                                   ;
                                  50 001C802C    8F  D0 00040 4$:  MOVL    #1867820, R0                                 ;
                                                     04 00047      RET                                                  ;
                                        0098     CB  D5 00048 5$:  TSTL    152(CTX)                                     ; 1614
                                                 04  12 0004C      BNEQ    6$                                           ;
                            5C    AB             04  88 0004E      BISB2   #4, 92(CTX)                                  ;
  0050  8F          00            6E             00  2C 00052 6$:  MOVC5   #0, (SP), #0, #80, FAB                       ; 1620
                                            10   AE     00059                                                           ;
                            10    AE      5003   8F  B0 0005B      MOVW    #20483, FAB                                  ; 1622
                                  58      009C   CB  D0 00061      MOVL    156(CTX), DDB                                ; 1627
                                  6E        59   AB  9A 00066      MOVZBL  89(CTX), I                                   ; 1628
                                          012E   31  0006A        BRW     21$                                          ;
                          00A0    CB             58  D0 0006D 7$:  MOVL    DDB, 160(CTX)                                ; 1633
                                  57        14   A8  9E 00072      MOVAB   20(R8), RAB                                  ; 1634
                                  50      0082   CB  9A 00076      MOVZBL  130(CTX), R0                                 ; 1643
        50          00            6E             00  2C 0007B      MOVC5   #0, (SP), #0, R0, @140(CTX)                  ;
                                        008C     DB     00080                                                           ;
                            2C    A7      008C   CB  D0 00083      MOVL    140(CTX), 44(RAB)                            ; 1651
                                        0081     CB  95 00089      TSTB    129(CTX)                                     ; 1652
                                                 03  12 0008D      BNEQ    8$                                           ;
                                          2C     A7  D4 0008F      CLRL    44(RAB)                                      ;
                                                 57  DD 00092 8$:  PUSHL   RAB                                          ; 1662
                      00000000G   00             01  FB 00094      CALLS   #1, SYS$GET                                  ;
                                  0E             50  E8 0009B      BLBS    R0, 9$                                       ;
                      000181A8    8F             50  D1 0009E      CMPL    R0, #98728                                   ; 1663
                                                 74  12 000A5      BNEQ    14$                                          ;
                            22    A7        0C   A7  B0 000A7      MOVW    12(RAB), 34(RAB)                             ; 1664
                                            7C   AB  D6 000AC 9$:  INCL    124(CTX)                                     ; 1676
                            04    AE        22   A7  3C 000AF      MOVZWL  34(RAB), D                                   ; 1681
                            08    AE        28   A7  D0 000B4      MOVL    40(RAB), D+4                                 ; 1682
                          0084    CB        04   AE  B1 000B9      CMPW    D, 132(CTX)                                  ; 1688
                                                 23  1B 000BF      BLEQU   10$                                          ;
```

```
                        04   A8  9F 000C1          PUSHAB  4(DDB)                          ; 1695
                             01  DD 000C4          PUSHL   #1                              ;
                  001C10B2   8F  DD 000C6          PUSHL   #1839282                        ;
                        10   AE  DD 000CC          PUSHL   D                               ;
                             01  DD 000CF          PUSHL   #1                              ;
                  001C8082   8F  DD 000D1          PUSHL   #1867906                        ;
        F84F   CF            06  FB 000D7          CALLS   #6, SOR$$ERROR                  ;
          04   AE      0084  CB  B0 000DC          MOVW    132(CTX), D                     ; 1696
                             08  11 000E2          BRB     11$                             ;
        0086   CB        04  AE  B1 000E4 10$:     CMPW    D, 134(CTX)                     ; 1700
                             0E  1F 000EA          BLSSU   13$                             ;
                        04   AE  9F 000EC 11$:     PUSHAB  D                               ; 1707
                  00000000G  00  16 000EF          JSB     SOR$$TREE_INSERT                ;
              5E             04  C0 000F5          ADDL2   #4, SP                          ;
                             98  11 000F8 12$:     BRB     8$                              ; 1683
                        04   A8  9F 000FA 13$:     PUSHAB  4(DDB)                          ; 1718
                             01  DD 000FD          PUSHL   #1                              ;
                  001C10B2   8F  DD 000FF          PUSHL   #1839282                        ;
                        10   AE  DD 00105          PUSHL   D                               ;
                             01  DD 00108          PUSHL   #1                              ;
                  001C80F8   8F  DD 0010A          PUSHL   #1868024                        ;
        F816   CF            06  FB 00110          CALLS   #6, SOR$$ERROR                  ;
                      00BC   CB  D6 00115          INCL    188(CTX)                        ; 1719
                             3C  11 00119          BRB     16$                             ; 1683
    000182DA   8F            50  D1 0011B 14$:     CMPL    R0, #99034                      ; 1723
                             0B  12 00122          BNEQ    15$                             ;
                             57  DD 00124          PUSHL   RAB                             ; 1725
    00000000G  00            01  FB 00126          CALLS   #1, SYS$WAIT                    ;
                             C9  11 0012D          BRB     12$                             ;
    000182AA   8F            50  D1 0012F 15$:     CMPL    R0, #98986                      ; 1727
                             21  12 00136          BNEQ    17$                             ;
1C        04   A7            03  E2 00138          BBSS    #3, 4(RAB), 17$                 ; 1732
                  00018029   8F  DD 0013D          PUSHL   #98345                          ; 1735
              7E        08   A7  7D 00143          MOVQ    8(RAB), -(SP)                   ;
                        04   A8  9F 00147          PUSHAB  4(DDB)                          ;
                             01  DD 0014A          PUSHL   #1                              ;
                  001C10B0   8F  DD 0014C          PUSHL   #1839280                        ;
        F7D4   CF            06  FB 00152          CALLS   #6, SOR$$ERROR                  ;
                             9F  11 00157 16$:     BRB     12$                             ; 1725
    0001827A   8F            50  D1 00159 17$:     CMPL    R0, #98938                      ; 1746
                             14  13 00160          BEQL    18$                             ;
              7E        08   A7  7D 00162          MOVQ    8(RAB), -(SP)                   ; 1749
                        04   A8  9F 00166          PUSHAB  4(DDB)                          ; 1748
                             01  DD 00169          PUSHL   #1                              ;
                  001C10B2   8F  DD 0016B          PUSHL   #1839282                        ;
        F7B5   CF            05  FB 00171          CALLS   #5, SOR$$ERROR                  ;
              02        58   AB  91 00176 18$:     CMPB    88(CTX), #2                     ; 1756
                             12  13 0017A          BEQL    19$                             ;
                  001C1052   8F  DD 0017C          PUSHL   #1839186                        ; 1758
                             58  DD 00182          PUSHL   DDB                             ;
                        18   AE  9F 00184          PUSHAB  FAB                             ;
        F880   CF            03  FB 00187          CALLS   #3, CLOSE_FILE                  ;
                             0A  11 0018C          BRB     20$                             ;
          32   A8            02  90 0018E 19$:     MOVB    #2, 50(DDB)                     ; 1765
          40   A8      008C  CB  D0 00192          MOVL    140(CTX), 64(DDB)               ; 1766
              58             68  D0 00198 20$:     MOVL    (DDB), DDB                      ; 1772
              02             6E  F4 0019B 21$:     SOBGEQ  I, 22$                          ; 1628
```

```
                                  03  11 0019E          BRB     23$                                  ;
                                FECA  31 001A0 22$:     BRW     7$                                   ;
          0000V  CF               00  FB 001A3 23$:     CALLS   #0, MRG_OUTPUT                       ; 1785
            0C   BE               01  8A 001A8          BICB2   #1, @CTX__                           ; 1788
            5C   AB               10  8A 001AC          BICB2   #16, 92(CTX)                         ;
                 50      0130     CB  D0 001B0          MOVL    304(CTX), R0                         ;
                                      04 001B5          RET                                          ; 1789
                                    0000 001B6 24$:     .WORD   Save nothing                         ; 1601
                 50        08     AC  D0 001B8          MOVL    8(AP), R0                            ;
                 50        04     A0  D0 001BC          MOVL    4(R0), R0                            ;
                            AC     A0  9F 001C0          PUSHAB  CTX__                                ;
                                   01  DD 001C3          PUSHL   #1                                   ;
                                   5E  DD 001C5          PUSHL   SP                                   ;
                 7E        04     AC  7D 001C7          MOVQ    4(AP), -(SP)                         ;
          F672   CF               03  FB 001CB          CALLS   #3, COND_HAND                        ;
                                      04 001D0          RET                                          ;

; Routine Size:  465 bytes,    Routine Base:  SOR$RO_CODE + 07DC
```

```
;  1737     1790  1 %IF NOT HOSTILE %THEN
;  1738     1791  1 ROUTINE MRG_OUTPUT:     CAL_CTXREG NOVALUE =
;  1739     1792  1
;  1740     1793  1 !++
;  1741     1794  1 !
;  1742     1795  1 ! FUNCTIONAL DESCRIPTION:
;  1743     1796  1 !
;  1744     1797  1 !       This routine merges into the output file.
;  1745     1798  1 !
;  1746     1799  1 ! FORMAL PARAMETERS:
;  1747     1800  1 !
;  1748     1801  1 !       NONE
;  1749     1802  1 !
;  1750     1803  1 ! IMPLICIT INPUTS:
;  1751     1804  1 !
;  1752     1805  1 !       NONE
;  1753     1806  1 !
;  1754     1807  1 ! IMPLICIT OUTPUTS:
;  1755     1808  1 !
;  1756     1809  1 !       NONE
;  1757     1810  1 !
;  1758     1811  1 ! ROUTINE VALUE:
;  1759     1812  1 !
;  1760     1813  1 !       Status code.
;  1761     1814  1 !
;  1762     1815  1 ! SIDE EFFECTS:
;  1763     1816  1 !
;  1764     1817  1 !       NONE
;  1765     1818  1 !
;  1766     1819  1 !--
;  1767     1820  2     BEGIN
;  1768     1821  2     EXTERNAL REGISTER
;  1769     1822  2         CTX =   COM_REG_CTX:    REF CTX_BLOCK;
;  1770     1823  2     LOCAL
;  1771     1824  2         DDB:    REF DDB_BLOCK,          ! Pointer to DDB
;  1772     1825  2         RAB:    REF BLOCK[,BYTE],       ! Pointer to RAB
;  1773     1826  2         FAB:    $FAB_DECL;              ! FAB (for closing files)
;  1774     1827  2
;  1775     1828  2
;  1776     1829  2     ! Check for record interface on output, RETURN_REC will be used
;  1777     1830  2     !
;  1778     1831  2     IF (DDB = .CTX[COM_OUT_DDB]) NEQ 0
;  1779     1832  2     THEN
;  1780     1833  3         BEGIN
;  1781     1834  3         LOCAL
;  1782     1835  3             D:  BLOCK[2];       ! Output buffer descriptor
;  1783     1836  3         MACRO
;  1784     1837  3             LENW = 0,0,16,0 %,
;  1785     1838  3             LENL = 0,0,32,0 %,
;  1786     1839  3             ADDR = 1,0,32,0 %;
;  1787     1840  3
;  1788     1841  3         ! Initialize FAB (used for closing files)
;  1789     1842  3         !
;  1790     1843  3         CH$FILL(0, FAB$C_BLN, FAB[BASE_]);
;  1791     1844  3         FAB[FAB$B_BLN] = FAB$C_BLN;
;  1792     1845  3         FAB[FAB$B_BID] = FAB$C_BID;
;  1793     1846  3
```

```
;  1794          1847  3
;  1795          1848  3          ! Get a pointer to the RAB
;  1796          1849  3          !
;  1797          1850  3          RAB = DDB[DDB_RAB+BASE_];
;  1798          1851  3
;  1799          1852  3
;  1800          1853  3          ! Set RAB$L_RHB to reference the VFC area
;  1801          1854  3          !
;  1802          1855  3          RAB[RAB$L_RHB] = .CTX[COM_RHB_OUT];
;  1803          1856  3
;  1804          1857  3
;  1805          1858  3          ! Allocate a record buffer area.
;  1806          1859  3          ! The length and address are stored in the USZ and UBF fields,
;  1807          1860  3          ! so that FREE_DDBS can release the memory.
;  1808          1861  3          !
;  1809          1862  3          RAB[RAB$W_USZ] = .CTX[COM_LRL_OUT];
;  1810          1863  3          RAB[RAB$L_UBF] = SOR$$ALLOCATE(.RAB[RAB$W_USZ]);
;  1811          1864  3
;  1812          1865  3          ! Initialize the descriptor for the returned record buffer.
;  1813          1866  3          ! Put the record address into the RAB, so RMS knows where it is.
;  1814          1867  3          ! The record length is written into the RAB by SOR$$TREE_EXTRACT.
;  1815          1868  3          !
;  1816          1869  3          D[LENL] = .CTX[COM_LRL_OUT];
;  1817          1870  3          D[ADDR] = RAB[RAB$L_RBF] = .RAB[RAB$L_UBF];
;  1818          1871  3
;  1819          1872  3
;  1820          1873  3          ! Get all the records from the sort, and write them to the output file.
;  1821          1874  3          !
;  1822          1875  4          BEGIN
;  1823          1876  4          BUILTIN R0;
;  1824          1877  4          WHILE TRUE DO
;  1825          1878  5              BEGIN
;  1826          1879  5
;  1827          1880  5              CHECKPOINT_COUNTDOWN();
;  1828          1881  5
;  1829          1882  5              ! Get the next record from the final merge pass.
;  1830          1883  5              !
;  1831          1884  5              R0 = SOR$$TREE_EXTRACT(D[BASE_], RAB[RAB$W_RSZ]);
;  1832          1885  5              IF NOT .R0 THEN EXITLOOP;
;  1833          1886  5
;  1834          1887  5
;  1835          1888  5              ! Output one record
;  1836          1889  5              !
;  1837          1890  5              WHILE TRUE DO
;  1838          1891  6                  BEGIN
;  1839          1892  7                  IF (R0 = $PUT(RAB=RAB[BASE_]))
;  1840          1893  6                  THEN
;  1841          1894  7                      BEGIN
;  1842          1895  7                      CTX[COM_OUT_RECNUM] = .CTX[COM_OUT_RECNUM] + 1;
;  1843          1896  7                      EXITLOOP                    ! Don't try again
;  1844          1897  7                      END
;  1845          1898  6                  ELIF
;  1846          1899  6                      .R0 EQL RMS$_RSZ
;  1847          1900  6                  THEN
;  1848          1901  7                      BEGIN
;  1849          1902  7                      !
;  1850          1903  7                      ! Complain about the length problem
```

```
;  1851     1904  7                    ! See if we have a better guess at the correct length.
;  1852     1905  7                    ! If not, drop this record from the sort.
;  1853     1906  7                    ! If so, try using it.
;  1854     1907  7                    !
;  1855     1908  7                    R0 = SOR$$ERROR(SOR$_SHR_WRITEERR, 1, DDB[DDB_NAME],
;  1856     1909  7                        .RAB[RAB$L_STS], .RAB[RAB$L_STV]);
;  1857     1910  7                    IF .RAB[RAB$W_RSZ] EQL .D[LENW]
;  1858     1911  7                    THEN
;  1859     1912  7                        EXITLOOP                    ! Don't try again
;  1860     1913  7                    ELIF .RAB[RAB$W_RSZ] LSSU .D[LENW]
;  1861     1914  7                    THEN
;  1862     1915  8                        BEGIN
;  1863     1916  8                        !
;  1864     1917  8                        ! Fill the record with the pad character.
;  1865     1918  8                        ! Note that this would be unnecessary if TREE_EXTRACT
;  1866     1919  8                        ! used COM_PAD.
;  1867     1920  8                        !
;  1868     1921  8                        BUILTIN MOVC5;
;  1869     1922  8                        MOVC5(%REF(0), .R0, CTX[COM_PAD],
;  1870     1923  8                            %REF(.D[LENW] - .RAB[RAB$W_RSZ]),
;  1871     1924  8                                (.D[ADDR] + .RAB[RAB$W_RSZ]); R0);
;  1872     1925  7                        END;
;  1873     1926  7                    RAB[RAB$W_RSZ] = .D[LENW];
;  1874     1927  7                    !
;  1875     1928  7                    ! The $PUT had better work this time, because
;  1876     1929  7                    ! D[LENW] = CTX[COM_LRL_OUT] = .OUTPUTFAB[FAB$W_MRS]
;  1877     1930  7                    !
;  1878     1931  7                    END                             ! And try again
;  1879     1932  6                ELIF
;  1880     1933  6                    .R0 EQL RMS$_RSA
;  1881     1934  6                THEN
;  1882     1935  7                    R0 = $WAIT(RAB=RAB[BASE_])  ! And try again
;  1883     1936  6                ELIF
;  1884     1937  6                    .R0 EQL RMS$_SEQ
;  1885     1938  6                THEN
;  1886     1939  7                    BEGIN
;  1887     1940  7                    R0 = SOR$$ERROR(SOR$_KEYED);
;  1888     1941  7                    RAB[RAB$B_RAC] = RAB$C_KEY; ! And try again
;  1889     1942  7                    END
;  1890     1943  6                ELSE
;  1891     1944  7                    BEGIN
;  1892     1945  7                    R0 = SOR$$ERROR(SOR$_SHR_WRITEERR, 1, DDB[DDB_NAME],
;  1893     1946  7                        .RAB[RAB$L_STS], .RAB[RAB$L_STV]);
;  1894     1947  7                    EXITLOOP;                       ! Don't try again
;  1895     1948  6                    END;
;  1896     1949  5                END;
;  1897     1950  4            END;
;  1898     1951  4
;  1899     1952  4
;  1900     1953  4        ! Check that we got the expected status
;  1901     1954  4        !
;  1902     1955  4        IF .R0 NEQ SS$_ENDOFFILE THEN SOR$$ERROR(.R0);
;  1903     1956  3        END;
;  1904     1957  3
;  1905     1958  3
;  1906     1959  3        ! Close the output file
;  1907     1960  3        !
```

```
; 1908        1961  3          CLOSE_FILE(FAB[BASE_], DDB[BASE_], SOR$_SHR_CLOSEOUT);
; 1909        1962  3
; 1910        1963  3
; 1911        1964  3          ! If this was a tag sort, close the input files now.
; 1912        1965  3          !
; 1913        1966  3          IF .CTX[COM_SORT_TYPE] EQL TYP_K_TAG
; 1914        1967  3          THEN
; 1915        1968  4              BEGIN
; 1916        1969  4
; 1917        1970  4              DDB = .CTX[COM_INP_DDB];
; 1918        1971  4              DECR I FROM .CTX[COM_NUM_FILES]-1 TO 0 DO
; 1919        1972  5                  BEGIN
; 1920        1973  5                  !
; 1921        1974  5                  ! Close the input file
; 1922        1975  5                  !
; 1923        1976  5                  CLOSE_FILE(FAB[BASE_], DDB[BASE_], SOR$_SHR_CLOSEIN);
; 1924        1977  5
; 1925        1978  5                  ! Advance to next DDB
; 1926        1979  5                  !
; 1927        1980  5                  DDB = .DDB[DDB_NEXT];
; 1928        1981  4                  END;
; 1929        1982  3              END;
; 1930        1983  3
; 1931        1984  2          END;
; 1932        1985  2
; 1933        1986  1      END;


                                                                  .EXTRN  SYS$PUT

                                                  07FC 00000 MRG_OUTPUT:
                                                                  .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10          ; 1791
                                  5E       A8   AE  9E 00002      MOVAB   -88(SP), SP                               ;
                                  58     0098   CB  D0 00006      MOVL    152(CTX), DDB                             ; 1831
                                                01  12 0000B      BNEQ    1$                                        ;
                                                    04 0000D      RET                                               ;
0050  8F          00              6E           00  2C 0000E 1$:   MOVC5   #0, (SP), #0, #80, FAB                    ; 1843
                                           08   AE     00015                                                        ;
                          08      AE     5003   8F  B0 00017      MOVW    #20483, FAB                               ; 1845
                                  57       14   A8  9E 0001D      MOVAB   20(R8), RAB                               ; 1850
                          2C      A7     0090   CB  D0 00021      MOVL    144(CTX), 44(RAB)                         ; 1855
                          20      A7     008A   CB  B0 00027      MOVW    138(CTX), 32(RAB)                         ; 1862
                                  7E       20   A7  3C 0002D      MOVZWL  32(RAB), -(SP)                            ; 1863
                        F796      CF           01  FB 00031      CALLS   #1, SOR$$ALLOCATE                         ;
                          24      A7           50  D0 00036      MOVL    R0, 36(RAB)                               ;
                                  6E     008A   CB  3C 0003A      MOVZWL  138(CTX), D                               ; 1869
                                  50       24   A7  D0 0003F      MOVL    36(RAB), R0                               ; 1870
                          28      A7           50  D0 00043      MOVL    R0, 40(RAB)                               ;
                          04      AE           50  D0 00047      MOVL    R0, D+4                                   ;
                                           22   A7  9F 0004B 2$:   PUSHAB  34(RAB)                                   ; 1884
                                           04   AE  9F 0004E      PUSHAB  D                                         ;
                                  00000000G     00  16 00051      JSB     SOR$$TREE_EXTRACT                         ;
                                  5E           08  C0 00057      ADDL2   #8, SP                                    ;
                                  03           50  E8 0005A      BLBS    R0, 3$                                    ; 1885
                                             009C  31 0005D      BRW     10$                                       ;
                                                57  DD 00060 3$:   PUSHL   RAB                                       ; 1892
```

```
                00000000G  00          01  FB 00062        CALLS   #1, SYS$PUT                          ;
                           06          50  E9 00069        BLBC    R0, 4$                               ;
                                 00C0  CB  D6 0006C        INCL    192(CTX)                             ; 1895
                                       D9  11 00070        BRB     2$                                   ; 1894
                000186A4   8F          50  D1 00072 4$:    CMPL    R0, #100004                          ; 1899
                                       3C  12 00079        BNEQ    7$                                   ;
                           7E      08  A7  7D 0007B        MOVQ    8(RAB), -(SP)                        ; 1909
                                   04  A8  9F 0007F        PUSHAB  4(DDB)                               ; 1908
                                       01  DD 00082        PUSHL   #1                                   ;
                        001C10D2       8F  DD 00084        PUSHL   #1839314                             ;
                    F6CB   CF          05  FB 0008A        CALLS   #5, SOR$$ERROR                       ;
                           6E      22  A7  B1 0008F        CMPW    34(RAB), D                           ; 1910
                                       B6  13 00093        BEQL    2$                                   ;
                                       1A  1E 00095        BGEQU   5$                                   ; 1913
                           52          6E  3C 00097        MOVZWL  D, R2                                ; 1923
                           51      22  A7  3C 0009A        MOVZWL  34(RAB), R1                          ;
                           52          51  C2 0009E        SUBL2   R1, R2                               ;
                           51      22  A7  3C 000A1        MOVZWL  34(RAB), R1                          ; 1924
                           51      04  AE  C0 000A5        ADDL2   D+4, R1                              ;
  52     0101   CB         60          00  2C 000A9        MOVC5   #0, (R0), 257(CTX), R2, (R1)         ;
                                       61     000B0                                                     ;
                      22   A7          6E  B0 000B1 5$:    MOVW    D, 34(RAB)                           ; 1926
                                       A9  11 000B5 6$:    BRB     3$                                   ; 1897
                000182DA   8F          50  D1 000B7 7$:    CMPL    R0, #99034                           ; 1933
                                       0B  12 000BE        BNEQ    8$                                   ;
                                       57  DD 000C0        PUSHL   RAB                                  ; 1935
                00000000G  00          01  FB 000C2        CALLS   #1, SYS$WAIT                         ;
                                       95  11 000C9        BRB     3$                                   ;
                000186AC   8F          50  D1 000CB 8$:    CMPL    R0, #100012                          ; 1937
                                       11  12 000D2        BNEQ    9$                                   ;
                        001C80F2       8F  DD 000D4        PUSHL   #1868018                             ; 1940
                    F67B   CF          01  FB 000DA        CALLS   #1, SOR$$ERROR                       ;
                      1E   A7          01  90 000DF        MOVB    #1, 30(RAB)                          ; 1941
                                       D0  11 000E3        BRB     6$                                   ; 1935
                           7E      08  A7  7D 000E5 9$:    MOVQ    8(RAB), -(SP)                        ; 1946
                                   04  A8  9F 000E9        PUSHAB  4(DDB)                               ; 1945
                                       01  DD 000EC        PUSHL   #1                                   ;
                        001C10D2       8F  DD 000EE        PUSHL   #1839314                             ;
                    F661   CF          05  FB 000F4        CALLS   #5, SOR$$ERROR                       ;
                                     FF4F  31 000F9        BRW     2$                                   ; 1944
                00000870   8F          50  D1 000FC 10$:   CMPL    R0, #2160                            ; 1955
                                       07  13 00103        BEQL    11$                                  ;
                                       50  DD 00105        PUSHL   R0                                   ;
                    F64E   CF          01  FB 00107        CALLS   #1, SOR$$ERROR                       ;
                        001C105A       8F  DD 0010C 11$:   PUSHL   #1839194                             ; 1961
                                       58  DD 00112        PUSHL   DDB                                  ;
                                   10  AE  9F 00114        PUSHAB  FAB                                  ;
                    F71F   CF          03  FB 00117        CALLS   #3, CLOSE_FILE                       ;
                           02      58  AB  91 0011C        CMPB    88(CTX), #2                          ; 1966
                                       21  12 00120        BNEQ    14$                                  ;
                           58    009C  CB  D0 00122        MOVL    156(CTX), DDB                        ; 1970
                           52      59  AB  9A 00127        MOVZBL  89(CTX), I                           ; 1971
                                       13  11 0012B        BRB     13$                                  ;
                        001C1052       8F  DD 0012D 12$:   PUSHL   #1839186                             ; 1976
                                       58  DD 00133        PUSHL   DDB                                  ;
                                   10  AE  9F 00135        PUSHAB  FAB                                  ;
                    F6FE   CF          03  FB 00138        CALLS   #3, CLOSE_FILE                       ;
```

```
                    58          68  D0 0013D          MOVL    (DDB), DDB                          ; 1980
                    EA          52  F4 00140 13$:     SOBGEQ  I, 12$                              ; 1971
                                    04 00143 14$:     RET                                         ; 1986

; Routine Size:  324 bytes,    Routine Base:  SOR$RO_CODE + 09AD

; 1934          1987  1 %FI
```

```
;1936     1988  1 GLOBAL ROUTINE SOR$RELEASE_REC                    ! Finished
;1937     1989  1         (
;1938     1990  1         DESC:           REF BLOCK[,BYTE],
;1939     1991  1         CONTEXT:        REF VECTOR[1,LONG]      ! Addr of context longword
;1940     1992  1         ) =
;1941     1993  1 !++
;1942     1994  1 !
;1943     1995  1 ! FUNCTIONAL DESCRIPTION:
;1944     1996  1 !
;1945     1997  1 !       Pass a record to the sort.
;1946     1998  1 !
;1947     1999  1 ! FORMAL PARAMETERS:
;1948     2000  1 !
;1949     2001  1 !       DESC.rab.d              Descriptor of the record
;1950     2002  1 !       CONTEXT.ra.r            Longword pointing to work area
;1951     2003  1 !
;1952     2004  1 !       All parameters except DESC are optional.
;1953     2005  1 !
;1954     2006  1 ! IMPLICIT INPUTS:
;1955     2007  1 !
;1956     2008  1 !       NONE
;1957     2009  1 !
;1958     2010  1 ! IMPLICIT OUTPUTS:
;1959     2011  1 !
;1960     2012  1 !       NONE
;1961     2013  1 !
;1962     2014  1 ! ROUTINE VALUE:
;1963     2015  1 !
;1964     2016  1 !       Status code.
;1965     2017  1 !
;1966     2018  1 ! SIDE EFFECTS:
;1967     2019  1 !
;1968     2020  1 !       NONE
;1969     2021  1 !
;1970     2022  1 !--
;1971     2023  2     BEGIN
;1972     2024  2     FIRSTPARAMETER_(DESC);      ! Required by PRESENT_ and NULL_ macros
;1973     2025  2     LOCAL
;1974     2026  2         STATUS;
;1975     2027  2
;1976     2028  2     CONTEXT_(CONTEXT, SOR$_SORT_ON);
;1977     2029  2
;1978     2030  2
;1979     2031  2     ! Check that we are allowed to get here, and check for DESC present
;1980     2032  2     !
;1981     2033  2     IF NOT .CTX[COM_FLO_RELEASE] THEN RETURN_(SOR$_SORT_ON);
;1982     2034  2     IF NULL_(DESC) THEN RETURN_(SOR$_MISS_PARAM);
;1983     2035  2
;1984     2036  2
;1985     2037  2     CTX[COM_INP_RECNUM] = .CTX[COM_INP_RECNUM] + 1;
;1986     2038  2     CHECKPOINT_COUNTDOWN();
;1987     2039  2
;1988     2040  2
;1989     2041  2     IF .DESC[DSC$W_LENGTH] LSSU .CTX[COM_SRL]
;1990     2042  2     THEN
;1991     2043  3         BEGIN
;1992     2044  3         !
```

```
;  1993          2045  3          ! If the record is too short, complain and ignore it
;  1994          2046  3          !
;  1995          2047  3          CTX[COM_OMI_RECNUM] = .CTX[COM_OMI_RECNUM] + 1;
;  1996          2048  3          RETURN__(SOR$$ERROR(SOR$_BAD_SRL));
;  1997          2049  3          END
;  1998          2050  2      ELIF .DESC[DSC$W_LENGTH] GTRU .CTX[COM_LRL]
;  1999          2051  2      THEN
;  2000          2052  2          BEGIN
;  2001          2053  2          !
;  2002          2054  2          ! If the record is too long, complain and use a shorter record
;  2003          2055  2          !
;  2004          2056  2          LOCAL
;  2005          2057  2              D: VECTOR[2];
;  2006          2058  2          D[0] = .CTX[COM_LRL];
;  2007          2059  2          D[1] = .DESC[DSC$A_POINTER];
;  2008          2060  2          SOR$$ERROR(SOR$_BAD_LRL, 1, .DESC[DSC$W_LENGTH]);
;  2009          2061  2          SOR$$TREE_INSERT(D[0]);
;  2010          2062  2          END
;  2011          2063  2      ELSE
;  2012          2064  2          BEGIN
;  2013          2065  2          !
;  2014          2066  2          ! Insert the record into the sort tree (this also converts the record)
;  2015          2067  2          !
;  2016          2068  2          SOR$$TREE_INSERT(DESC[BASE_]);
;  2017          2069  2          END;
;  2018          2070  2
;  2019          2071  2      RETURN__(SS$_NORMAL);
;  2020          2072  1      END;


                                                OFFC 00000          .ENTRY  SOR$RELEASE_REC, Save R2,R3,R4,R5,R6,R7,R8,-; 1988
                                                                            R9,R10,R11
                                                                    ;
                                     5E         0C   C2 00002       SUBL2   #12, SP                                       ;
                                                0C   AE D4 00005       CLRL    CTX__                                      ; 2023
                                     6D   00B1  CF   DE 00008       MOVAL   11$, (FP)                                     ;
                                     02         6C   91 0000D       CMPB    (AP), #2                                   ; 2028
                                                06   1F 00010       BLSSU   1$                                            ;
                                     53         08   AC D0 00012       MOVL    CONTEXT, P__                               ;
                                                07   12 00016       BNEQ    2$                                         ;
                                     53 00000000' EF   9E 00018  1$:     MOVAB   SOR_CONTEXT, P__                           ;
                                08   AE         53   D0 0001F  2$:     MOVL    P__, CTX__                                 ;
                              19    63         00   E2 00023       BBSS    #0, (P__), 4$                                  ;
                              5B    63         01   CB 00027       BICL3   #1, (P__), CTX                                 ;
                                               0F   13 0002B       BEQL    3$                                         ;
                              0A  5C   AB              04   E2 0002D       BBSS    #4, 92(CTX), 3$                            ;
                                01C0   CB              01   D0 00032       MOVL    #1, 304(CTX)                               ;
                              0C  5C   AB              01   E0 00037       BBS     #1, 92(CTX), 5$                            ; 2033
                                  08   BE              01   8A 0003C  3$:     BICB2   #1, @CTX__                         ;
                                     50 001C802C   8F   D0 00040  4$:     MOVL    #1867820, R0                               ;
                                                   04   00047       RET                                                    ;
                                                6C   95 00048  5$:     TSTB    (AP)                                       ; 2034
                                                05   13 0004A       BEQL    6$                                         ;
                                          04    AC   D5 0004C       TSTL    4(AP)                                      ;
                                                0C   12 0004F       BNEQ    7$                                         ;
```

```
        08  BE              01  8A 00051 6$:    BICB2   #1, @CTX__                  ;
            50 001C80E4     8F  D0 00055        MOVL    #1868004, R0                ;
                                04 0005C        RET                                 ;
                      7C    AB  D6 0005D 7$:    INCL    124(CTX)                    ; 2037
            50        04    AC  D0 00060        MOVL    DESC, R0                    ; 2041
      0086  CB              60  B1 00064        CMPW    (R0), 134(CTX)              ;
                            14  1E 00069        BGEQU   8$                          ;
                    00BC    CB  D6 0006B        INCL    188(CTX)                    ; 2047
        08  BE              01  8A 0006F        BICB2   #1, @CTX__                  ; 2048
                001C80F8    8F  DD 00073        PUSHL   #1868024__                  ;
      F598  CF              01  FB 00079        CALLS   #1, SOR$$ERROR              ;
                                04 0007E        RET                                 ;
      0084  CB              60  B1 0007F 8$:    CMPW    (R0), 132(CTX)              ; 2050
                            1E  1B 00084        BLEQU   9$                          ;
            6E      0084    CB  3C 00086        MOVZWL  132(CTX), D                 ; 2058
        04  AE        04    A0  D0 0008B        MOVL    4(R0), D+4                  ; 2059
            7E              60  3C 00090        MOVZWL  (R0), -(SP)                 ; 2060
                            01  DD 00093        PUSHL   #1                          ;
                001C8082    8F  DD 00095        PUSHL   #1867906                    ;
      F576  CF              03  FB 0009B        CALLS   #3, SOR$$ERROR              ;
                            5E  DD 000A0        PUSHL   SP                          ; 2061
                            02  11 000A2        BRB     10$                         ;
                            50  DD 000A4 9$:    PUSHL   R0                          ; 2068
                00000000G   00  16 000A6 10$:   JSB     SOR$$TREE_INSERT            ;
            5E              04  C0 000AC        ADDL2   #4, SP                      ;
        08  BE              01  8A 000AF        BICB2   #1, @CTX__                  ; 2071
        5C  AB              10  8A 000B3        BICB2   #16, 92(CTX)                ;
            50      0130    CB  D0 000B7        MOVL    304(CTX), R0                ;
                                04 000BC        RET                                 ; 2072
                            0000 000BD 11$:     .WORD   Save nothing                ; 2023
            50        08    AC  D0 000BF        MOVL    8(AP), R0                   ;
            50        04    A0  D0 000C3        MOVL    4(R0), R0                   ;
                      FC    A0  9F 000C7        PUSHAB  CTX__                       ;
                            01  DD 000CA        PUSHL   #1                          ;
                            5E  DD 000CC        PUSHL   SP                          ;
            7E        04    AC  7D 000CE        MOVQ    4(AP), -(SP)                ;
      F456  CF              03  FB 000D2        CALLS   #3, COND_HAND               ;
                                04 000D7        RET                                 ;

; Routine Size:  216 bytes,    Routine Base:  SOR$RO_CODE + 0AF1
```

```
;  2022     2073  1 GLOBAL ROUTINE SOR$RETURN_REC                      ! Finished
;  2023     2074  1         (
;  2024     2075  1         DESC:           REF BLOCK[,BYTE],          ! Buffer descriptor
;  2025     2076  1         SIZE:           REF VECTOR[1,WORD],        ! Output length
;  2026     2077  1         CONTEXT:        REF VECTOR[1,LONG]         ! Addr of context longword
;  2027     2078  1         ) =
;  2028     2079  1 !++
;  2029     2080  1 !
;  2030     2081  1 ! FUNCTIONAL DESCRIPTION:
;  2031     2082  1 !
;  2032     2083  1 !       Get a record returned from the sort.
;  2033     2084  1 !
;  2034     2085  1 ! FORMAL PARAMETERS:
;  2035     2086  1 !
;  2036     2087  1 !       DESC.wab.d              Descriptor of the record
;  2037     2088  1 !       SIZE.waw.r              Word to recieve length of record
;  2038     2089  1 !       CONTEXT.ra.r            Longword pointing to work area
;  2039     2090  1 !
;  2040     2091  1 !       All parameters except DESC are optional.
;  2041     2092  1 !
;  2042     2093  1 ! IMPLICIT INPUTS:
;  2043     2094  1 !
;  2044     2095  1 !       NONE
;  2045     2096  1 !
;  2046     2097  1 ! IMPLICIT OUTPUTS:
;  2047     2098  1 !
;  2048     2099  1 !       NONE
;  2049     2100  1 !
;  2050     2101  1 ! ROUTINE VALUE:
;  2051     2102  1 !
;  2052     2103  1 !       Status code.
;  2053     2104  1 !
;  2054     2105  1 ! SIDE EFFECTS:
;  2055     2106  1 !
;  2056     2107  1 !       NONE
;  2057     2108  1 !
;  2058     2109  1 !--
;  2059     2110  2     BEGIN
;  2060     2111  2     FIRSTPARAMETER_(DESC);      ! Required by PRESENT_ and NULL_ macros
;  2061     2112  2     LOCAL
;  2062     2113  2         STATUS;
;  2063     2114  2
;  2064     2115  2     CONTEXT_(CONTEXT, SOR$_SORT_ON);
;  2065     2116  2
;  2066     2117  2
;  2067     2118  2     ! Check that we are allowed to get here, and check for DESC present
;  2068     2119  2     !
;  2069     2120  2     IF NOT .CTX[COM_FLO_RETURN] THEN RETURN_(SOR$_SORT_ON);
;  2070     2121  2     IF NULL_(DESC) THEN RETURN_(SOR$_MISS_PARAM);
;  2071     2122  2
;  2072     2123  2
;  2073     2124  2     CHECKPOINT_COUNTDOWN();
;  2074     2125  2
;  2075     2126  2
;  2076     2127  2     ! Call TREE_EXTRACT to get the record from the tree (this also converts
;  2077     2128  2     ! the record to the output format).
;  2078     2129  2     !
```

```
;  2079          2130  2     STATUS = SOR$$TREE_EXTRACT(DESC[BASE_],
;  2080          2131  2         (IF PRESENT_(SIZE) THEN SIZE[0] ELSE 0));
;  2081          2132  2
;  2082          2133  2
;  2083          2134  2     ! If there are no more records, the user isn't supposed to call us again
;  2084          2135  2     !
;  2085          2136  2     IF .STATUS EQL SS$_ENDOFFILE                ! Check for the end
;  2086          2137  2     THEN
;  2087          2138  3         BEGIN
;  2088          2139  3         !
;  2089          2140  3         ! Note that further calls to RETURN_REC will also return SS$_ENDOFFILE
;  2090          2141  3         !
;  2091          2142  3         CTX[COM_FLO_ABORT] = FALSE;              ! Can call other routines
;  2092          2143  3         RETURN__(.STATUS);
;  2093          2144  3         END
;  2094          2145  2     ELSE
;  2095          2146  2         RETURN__(SS$_NORMAL);
;  2096          2147  2
;  2097          2148  1     END;


                                    0FFC 00000          .ENTRY  SOR$RETURN_REC, Save R2,R3,R4,R5,R6,R7,R8,- ; 2073
                                                                R9,R10,R11
                                7E  D4 00002            CLRL    CTX__                                       ; 2110
                          6D    CF  DE 00004            MOVAL   11$, (FP)
                          03        6C 91 00009         CMPB    (AP), #3                                    ; 2115
                                    06 1F 0000C         BLSSU   1$
                          53    0C  AC D0 0000E         MOVL    CONTEXT, P__
                                    07 12 00012         BNEQ    2$
                          53 00000000'  EF 9E 00014 1$: MOVAB   SOR_CONTEXT, P__
                          6E        53 D0 0001B 2$:     MOVL    P__, CTX__
                    19    63        00 E2 0001E         BBSS    #0, (P__), 4$
                    5B    63        01 CB 00022         BICL3   #1, (P__), CTX
                                    0F 13 00026         BEQL    3$
                    0A  5C  AB          04 E2 00028     BBSS    #4, 92(CTX), 3$
                  0130  CB              01 D0 0002D     MOVL    #1, 304(CTX)
                    0C  5C  AB          02 E0 00032     BBS     #2, 92(CTX), 5$                             ; 2120
                        00  BE          01 8A 00037 3$: BICB2   #1, @CTX__
                          50 001C802C   8F D0 0003B 4$: MOVL    #1867820, R0
                                        04 00042        RET
                                    6C 95 00043 5$:     TSTB    (AP)                                        ; 2121
                                    05 13 00045         BEQL    6$
                                04  AC D5 00047         TSTL    4(AP)
                                    0C 12 0004A         BNEQ    7$
                          00    BE      01 8A 0004C 6$: BICB2   #1, @CTX__
                          50 001C80E4   8F D0 00050     MOVL    #1868004, R0
                                        04 00057        RET
                          02            6C 91 00058 7$: CMPB    (AP), #2                                    ; 2131
                                        05 1F 0005B     BLSSU   8$
                                08      AC DD 0005D     PUSHL   SIZE
                                        02 11 00060     BRB     9$
                                    7E  D4 00062 8$:    CLRL    -(SP)
                                04      AC DD 00064 9$: PUSHL   DESC                                        ; 2130
                             00000000G  00 16 00067     JSB     SOR$$TREE_EXTRACT
```

```
                  5E          08  C0 0006D        ADDL2   #8, SP                          ;
        00000870  8F          50  D1 00070        CMPL    STATUS, #2160                   ; 2136
                              09  12 00077        BNEQ    10$                             ;
              5C  AB          10  8A 00079        BICB2   #16, 92(CTX)                    ; 2142
              00  BE          01  8A 0007D        BICB2   #1, @CTX__                      ; 2143
                                  04 00081        RET                                     ; 2146
              00  BE          01  8A 00082 10$:   BICB2   #1, @CTX__                      ;
              5C  AB          10  8A 00086        BICB2   #16, 92(CTX)                    ;
                  50    0130  CB  D0 0008A        MOVL    304(CTX), R0                    ;
                                  04 0008F        RET                                     ; 2148
                              0000 00090 11$:     .WORD   Save nothing                    ; 2110
                  50      08  AC  D0 00092        MOVL    8(AP), R0                       ;
                  50      04  A0  D0 00096        MOVL    4(R0), R0                       ;
                          FC  A0  9F 0009A        PUSHAB  CTX__                           ;
                              01  DD 0009D        PUSHL   #1                              ;
                              5E  DD 0009F        PUSHL   SP                              ;
                  7E      04  AC  7D 000A1        MOVQ    4(AP), -(SP)                    ;
            F3AB  CF          03  FB 000A5        CALLS   #3, COND_HAND                   ;
                                  04 000AA        RET                                     ;

; Routine Size:  171 bytes,    Routine Base:  SOR$RO_CODE + 0BC9
```

```
;2099     2149  1 %IF NOT HOSTILE %THEN
;2100     2150  1 ROUTINE FREE_DDBS                                       ! Finished
;2101     2151  1         (
;2102     2152  1         DDB_PTR:        REF VECTOR[1,LONG],            ! Address of pointer to DDBs
;2103     2153  1         CLOSEERR,                                      ! Error message
;2104     2154  1         DELUBF                                         ! Delete user buffer
;2105     2155  1         ):      CAL_CTXREG NOVALUE =
;2106     2156  1 !++
;2107     2157  1 !
;2108     2158  1 ! FUNCTIONAL DESCRIPTION:
;2109     2159  1 !
;2110     2160  1 !       Deallocate the storage associated with a list of DDBs.
;2111     2161  1 !
;2112     2162  1 !       NOTE:  The files are closed by this routine.
;2113     2163  1 !
;2114     2164  1 ! FORMAL PARAMETERS:
;2115     2165  1 !
;2116     2166  1 !       DDB_PTR.ra.r    Address of pointer to a list of DDBs
;2117     2167  1 !       CLOSEERR        Message to signal on close error
;2118     2168  1 !       DELUBF          True to delete the user buffer for these DDBs
;2119     2169  1 !
;2120     2170  1 ! IMPLICIT INPUTS:
;2121     2171  1 !
;2122     2172  1 !       NONE
;2123     2173  1 !
;2124     2174  1 ! IMPLICIT OUTPUTS:
;2125     2175  1 !
;2126     2176  1 !       NONE
;2127     2177  1 !
;2128     2178  1 ! ROUTINE VALUE:
;2129     2179  1 !
;2130     2180  1 !       NONE (signals errors)
;2131     2181  1 !
;2132     2182  1 ! SIDE EFFECTS:
;2133     2183  1 !
;2134     2184  1 !       NONE
;2135     2185  1 !
;2136     2186  1 !--
;2137     2187  2     BEGIN
;2138     2188  2     EXTERNAL REGISTER
;2139     2189  2         CTX =   COM_REG_CTX:    REF CTX_BLOCK;
;2140     2190  2     LOCAL
;2141     2191  2         FAB:    $FAB_DECL,
;2142     2192  2         DDB:    REF DDB_BLOCK,
;2143     2193  2         DDB_ADR:REF VECTOR[1],
;2144     2194  2         STATUS;
;2145     2195  2
;2146     2196  2
;2147     2197  2     ! Initialize a FAB
;2148     2198  2     !
;2149     2199  2     $FAB_INIT(FAB = FAB[BASE_]);
;2150     2200  2
;2151     2201  2
;2152     2202  2     ! Deallocate the user buffer if requested (and it is there)
;2153     2203  2     !
;2154     2204  2     DDB = .DDB_PTR[0];                                 ! Pointer to first DDB
;2155     2205  2     IF DDB[BASE_] NEQ 0 AND .DELUBF
```

```
;  2156          2206  2      THEN                                    ! Deallocate the user buffer
;  2157          2207  3          BEGIN
;  2158          2208  3          WHILE (DDB = .DDB[DDB_NEXT]) NEQ 0 DO DDB[DDB_RAB+RAB$L_UBF] = 0;
;  2159          2209  3          DDB = .DDB_PTR[0];                  ! Pointer to first DDB
;  2160          2210  3          SOR$$DEALLOCATE(.DDB[DDB_RAB+RAB$W_USZ], DDB[DDB_RAB+RAB$L_UBF]);
;  2161          2211  3          END;
;  2162          2212  2
;  2163          2213  2
;  2164          2214  2      ! For each DDB linked into this list
;  2165          2215  2      !
;  2166          2216  2      DDB_ADR = DDB_PTR[0];
;  2167          2217  2      WHILE (DDB = .DDB_ADR[0]) NEQ 0 DO
;  2168          2218  3          BEGIN
;  2169          2219  3
;  2170          2220  3          ! Close the file
;  2171          2221  3          !
;  2172          2222  3          IF .DDB[DDB_IFI] NEQ 0
;  2173          2223  3          THEN
;  2174          2224  3              CLOSE_FILE(FAB[BASE_], DDB[BASE_], .CLOSEERR);
;  2175          2225  3
;  2176          2226  3          ! Free the file name string
;  2177          2227  3          !
;  2178          2228  3          SOR$$FREE_FILE_NAME(DDB[DDB_NAME]);     ! Free name string
;  2179          2229  3
;  2180          2230  3          ! Free the DDB
;  2181          2231  3          !
;  2182          2232  3          SOR$$DEALLOCATE(DDB_K_SIZE, DDB_ADR[0], .DDB[DDB_NEXT]);
;  2183          2233  3
;  2184          2234  2          END;
;  2185          2235  2
;  2186          2236  1      END;


                                        003C 00000 FREE_DDBS:
                                                   .WORD   Save R2,R3,R4,R5                                    ; 2150
                    5E        B0  AE    9E 00002           MOVAB   -80(SP), SP                                 ;
    0050  8F              00    6E    00 2C 00006          MOVC5   #0, (SP), #0, #80, $RMS_PTR                 ; 2199
                                  6E       0000D           
                    6E      5003  8F    B0 0000E           MOVW    #20483, $RMS_PTR                            ;
                 16 AE            02    90 00013           MOVB    #2, $RMS_PTR+22                             ;
                 1F AE            02    90 00017           MOVB    #2, $RMS_PTR+31                             ;
                    52        04  BC    D0 0001B           MOVL    @DDB_PTR, DDB                               ; 2204
                                  1E    13 0001F           BEQL    3$                                          ; 2205
                    1A        0C  AC    E9 00021           BLBC    DELUBF, 3$                                  ;
                    52            62    D0 00025 1$:       MOVL    (DDB), DDB                                  ; 2208
                                  05    13 00028           BEQL    2$                                          ;
                            38    A2    D4 0002A           CLRL    56(DDB)                                     ;
                                  F6    11 0002D           BRB     1$                                          ;
                    52        04  BC    D0 0002F 2$:       MOVL    @DDB_PTR, DDB                               ; 2209
                            38    A2    9F 00033           PUSHAB  56(DDB)                                     ; 2210
                    7E      34    A2    3C 00036           MOVZWL  52(DDB), -(SP)                              ;
                F4F5        CF          02 FB 0003A        CALLS   #2, SOR$$DEALLOCATE                         ;
                    53        04  AC    D0 0003F 3$:       MOVL    DDB_PTR, DDB_ADR                            ; 2216
                    52            63    D0 00043 4$:       MOVL    (DDB_ADR), DDB                              ; 2217
```

```
                                  2B  13 00046          BEQL    6$                                  ;
                            0C    A2  D5 00048          TSTL    12(DDB)                             ; 2222
                                  0D  13 0004B          BEQL    5$                                  ;
                            08    AC  DD 0004D          PUSHL   CLOSEERR                            ; 2224
                                  52  DD 00050          PUSHL   DDB                                 ;
                            08    AE  9F 00052          PUSHAB  FAB                                 ;
                F51A   CF         03  FB 00055          CALLS   #3, CLOSE_FILE                      ;
                            04    A2  9F 0005A 5$:      PUSHAB  4(DDB)                              ; 2228
          00000000G   00          01  FB 0005D          CALLS   #1, SOR$$FREE_FILE_NAME             ;
                                  62  DD 00064          PUSHL   (DDB)                               ; 2232
                                  53  DD 00066          PUSHL   DDB_ADR                             ;
                      7E    59    8F  9A 00068          MOVZBL  #89, -(SP)                          ;
                F4C3   CF         03  FB 0006C          CALLS   #3, SOR$$DEALLOCATE                 ;
                                  D0  11 00071          BRB     4$                                  ; 2217
                                      04 00073 6$:      RET                                         ; 2236

; Routine Size:  116 bytes,    Routine Base:  SOR$RO_CODE + 0C74

; 2187          2237  1 %FI
```

```
;  2189          2238  1 ROUTINE AST_END_SORT
;  2190          2239  1         (
;  2191          2240  1         CTXADR: REF VECTOR[1]
;  2192          2241  1         ): JSB_INI_CTX =
;  2193          2242  1
;  2194          2243  1 !++
;  2195          2244  1 !
;  2196          2245  1 ! FUNCTIONAL DESCRIPTION:
;  2197          2246  1 !
;  2198          2247  1 !       This routine aborts a sort while another sort routine is active.
;  2199          2248  1 !
;  2200          2249  1 ! FORMAL PARAMETERS:
;  2201          2250  1 !
;  2202          2251  1 !       NONE
;  2203          2252  1 !
;  2204          2253  1 ! IMPLICIT INPUTS:
;  2205          2254  1 !
;  2206          2255  1 !       NONE
;  2207          2256  1 !
;  2208          2257  1 ! IMPLICIT OUTPUTS:
;  2209          2258  1 !
;  2210          2259  1 !       NONE
;  2211          2260  1 !
;  2212          2261  1 ! ROUTINE VALUE:
;  2213          2262  1 !
;  2214          2263  1 !       Status code
;  2215          2264  1 !
;  2216          2265  1 ! SIDE EFFECTS:
;  2217          2266  1 !
;  2218          2267  1 !       NONE
;  2219          2268  1 !
;  2220          2269  1 !--
;  2221          2270  2     BEGIN
;  2222          2271  2     EXTERNAL REGISTER
;  2223          2272  2         CTX =   COM_REG_CTX:    REF CTX_BLOCK;
;  2224          2273  2
;  2225        L 2274  2     %IF NOT SOR_K_END_AST
;  2226        U 2275  2     %THEN
;  2227        U 2276  2         RETURN SOR$_SORT_ON
;  2228          2277  2     %ELSE
;  2229          2278  3         BEGIN
;  2230          2279  3         EXTERNAL ROUTINE
;  2231          2280  3             LIB$AST_IN_PROG:    ADDRESSING_MODE(GENERAL),       ! At AST?
;  2232          2281  3             SYS$CLRAST:         ADDRESSING_MODE(GENERAL);       ! Clear AST
;  2233          2282  3         LOCAL
;  2234          2283  3             STATUS;
;  2235          2284  3
;  2236          2285  3         ROUTINE END_SORT_AST(CTXADR: REF VECTOR[1]): NOVALUE =
;  2237          2286  4             BEGIN
;  2238          2287  4             SYS$CLRAST();
;  2239          2288  4             !
;  2240          2289  4             ! Note that we should never return from this signal.
;  2241          2290  4             !
;  2242          2291  4             LIB$SIGNAL(SOR$_END_SORT, 1, CTXADR[0]);
;  2243          2292  3             END;
```

```
                                                                .EXTRN  LIB$AST_IN_PROG
                                                                .EXTRN  SYS$CLRAST

                                        0000 00000 END_SORT_AST:
                                                                .WORD   Save nothing                          ; 2285
        00000000G  00                 00  FB 00002              CALLS   #0, SYS$CLRAST                        ; 2287
                                 04   AC  DD 00009              PUSHL   CTXADR                                ; 2291
                                      01  DD 0000C              PUSHL   #1                                    ;
                          001C80E9    8F  DD 0000E              PUSHL   #1868009                              ;
        00000000G  00                 03  FB 00014              CALLS   #3, LIB$SIGNAL                        ;
                                          04 0001B              RET                                           ; 2292

; Routine Size:  28 bytes,    Routine Base:  SOR$RO_CODE + 0CE8


;  2244          2293  3
;  2245          2294  3          IF NOT LIB$AST_IN_PROG()
;  2246          2295  3          THEN
;  2247          2296  4              BEGIN
;  2248          2297  4              !
;  2249          2298  4              ! We were not called from an AST.
;  2250          2299  4              ! Just signal an abort request to the active sort/merge routine.
;  2251          2300  4              ! Note that we call LIB$SIGNAL directly so that when this signal
;  2252          2301  4              ! is caught by END_SORT's condition handler, the depth is zero.
;  2253          2302  4              ! Note that we should never return from this signal.
;  2254          2303  4              !
;  2255          2304  4              LIB$SIGNAL(SOR$_END_SORT, 1, CTXADR[0]);
;  2256          2305  3              END;
;  2257          2306  3
;  2258          2307  3          ! We were called from an AST routine.
;  2259          2308  3          ! Declare another AST routine so the current one can finish executing.
;  2260          2309  3          !
;  2261        P 2310  3          STATUS = $DCLAST(ASTADR=END_SORT_AST, ASTPRM=CTXADR[0]
;  2262          2311  3              %IF HOSTILE_ELAN %THEN , ELAN = .CTX[COM_CTXADR] %FI );
;  2263          2312  3          IF NOT .STATUS THEN RETURN SOR$$ERROR(.STATUS);
;  2264          2313  3
;  2265          2314  3          ! Return an informational status indicating that, although the caller
;  2266          2315  3          ! requested us to abort, we have not yet done so.
;  2267          2316  3          !
;  2268          2317  3          RETURN SOR$_END_SORT AND NOT STS$M_SEVERITY OR STS$K_INFO;
;  2269          2318  3          END
;  2270          2319  3
;  2271          2320  2      %FI;
;  2272          2321  2
;  2273          2322  1      END;


                                                                .EXTRN  SYS$DCLAST

        00000000G  00                 00  FB 00000 AST_END_SORT:
                                                                CALLS   #0, LIB$AST_IN_PROG                   ; 2294
                   11                 50  E8 00007              BLBS    R0, 1$                                ;
                                      53  DD 0000A              PUSHL   CTXADR                                ; 2304
                                      01  DD 0000C              PUSHL   #1                                    ;
                          001C80E9    8F  DD 0000E              PUSHL   #1868009                              ;
```

```
                00000000G  00          03  FB 00014          CALLS   #3, LIB$SIGNAL                  ;
                                       7E  D4 0001B 1$:      CLRL    -(SP)                           ; 2311
                                       53  DD 0001D          PUSHL   CTXADR                          ;
                                  C2   AF  9F 0001F          PUSHAB  END_SORT_AST                    ;
                00000000G  00          03  FB 00022          CALLS   #3, SYS$DCLAST                  ;
                           08          50  E8 00029          BLBS    STATUS, 2$                      ; 2312
                                       50  DD 0002C          PUSHL   STATUS                          ;
                    F3D0   CF          01  FB 0002E          CALLS   #1, SOR$$ERROR                  ;
                                           05 00033          RSB                                     ;
                           50 001C80EB 8F  D0 00034 2$:      MOVL    #1868011, R0                    ; 2317
                                           05 0003B          RSB                                     ; 2322

; Routine Size:  60 bytes,    Routine Base:  SOR$RO_CODE + 0D04
```

```
;  2275          2323  1 GLOBAL ROUTINE SOR$END_SORT
;  2276          2324  1         (
;  2277          2325  1         CONTEXT:        REF VECTOR[1,LONG]      ! Addr of context longword
;  2278          2326  1         ) =
;  2279          2327  1 !++
;  2280          2328  1 !
;  2281          2329  1 ! FUNCTIONAL DESCRIPTION:
;  2282          2330  1 !
;  2283          2331  1 !       Indicate the end of the sort, and free all storage used by the sort.
;  2284          2332  1 !
;  2285          2333  1 ! FORMAL PARAMETERS:
;  2286          2334  1 !
;  2287          2335  1 !       CONTEXT.ra.r            Longword pointing to work area
;  2288          2336  1 !
;  2289          2337  1 !       All parameters are optional.
;  2290          2338  1 !
;  2291          2339  1 ! IMPLICIT INPUTS:
;  2292          2340  1 !
;  2293          2341  1 !       NONE
;  2294          2342  1 !
;  2295          2343  1 ! IMPLICIT OUTPUTS:
;  2296          2344  1 !
;  2297          2345  1 !       NONE
;  2298          2346  1 !
;  2299          2347  1 ! ROUTINE VALUE:
;  2300          2348  1 !
;  2301          2349  1 !       Status code.
;  2302          2350  1 !
;  2303          2351  1 ! SIDE EFFECTS:
;  2304          2352  1 !
;  2305          2353  1 !       The following data structures are deallocated.
;  2306          2354  1 !
;  2307          2355  1 !       The Common context area.
;  2308          2356  1 !       The DDBs.
;  2309          2357  1 !               For each DDB, the file name string.
;  2310          2358  1 !               The user buffer for the input DDBs.
;  2311          2359  1 !               The user buffer for the output DDB.
;  2312          2360  1 !       The WFBs.
;  2313          2361  1 !               For each WFB, the file name string.
;  2314          2362  1 !       The replacement selection tree.
;  2315          2363  1 !       The run description blocks.
;  2316          2364  1 !       The work file buffers.
;  2317          2365  1 !
;  2318          2366  1 !--
;  2319          2367  2     BEGIN
;  2320          2368  2     FIRSTPARAMETER_(CONTEXT);   ! Required by PRESENT_ and NULL_ macros
;  2321          2369  2
;  2322          2370  2     PSECT NODEFAULT = SOR$$END_PSECT_(1)(PIC,SHARE,NOWRITE,EXECUTE,
;  2323          2371  2                                         ADDRESSING_MODE(LONG_RELATIVE));
;  2324          2372  2     OWN END_0: VECTOR[0] PSECT(SOR$$END_PSECT_(1));
;  2325          2373  2     PSECT NODEFAULT = SOR$$END_PSECT_(3)(PIC,SHARE,NOWRITE,EXECUTE,
;  2326          2374  2                                         ADDRESSING_MODE(LONG_RELATIVE));
;  2327          2375  2     OWN END_1: VECTOR[0] PSECT(SOR$$END_PSECT_(3));
;  2328          2376  2
;  2329          2377  2     CONTEXT_(CONTEXT, SS$_NORMAL);
;  2330          2378  2
;  2331          2379  2 !+
```

```
;  2332          2380  2 !   Allow END_SORT to be called at any time.
;  2333          2381  2 !
;  2334          2382  2 !   IF NOT .CTX[COM_FLO_RETURN] THEN RETURN__(SOR$_SORT_ON);
;  2335          2383  2 !-
;  2336          2384  2
;  2337          2385  2       ! Call the clean-up routines
;  2338          2386  2       !
;  2339          2387  3       BEGIN
;  2340          2388  3       LOCAL
;  2341          2389  3           P: REF VECTOR[1],
;  2342          2390  3           Q: REF VECTOR[1];
;  2343          2391  3       P = END_0;
;  2344          2392  3       Q = END_1;
;  2345          2393  3       WHILE P[0] LSSA Q[0] DO
;  2346          2394  4           BEGIN
;  2347          2395  4           LOCAL RTN, STATUS;
;  2348          2396  4           RTN = .P[0] + P[0]; P = P[1];
;  2349          2397  4           CAL_CTXREG(.RTN);
;  2350          2398  3           END;
;  2351          2399  2       END;
;  2352          2400  2
;  2353          2401  2
;  2354        C 2402  2 %(
;  2355        C 2403  2       ! Unlock everything that got locked into the WS
;  2356        C 2404  2       !
;  2357        C 2405  2       BEGIN
;  2358        C 2406  2       LOCAL
;  2359        C 2407  2           P:      REF BLOCK;
;  2360        C 2408  2       P = .CTX[COM_LOCKED];
;  2361        C 2409  2       WHILE P[BASE_] NEQ 0 DO
;  2362        C 2410  2           BEGIN
;  2363        C 2411  2           P = .P[LCK_NEXT];
;  2364        C 2412  2           END;
;  2365        C 2413  2       END;
;  2366          2414  2 )%
;  2367          2415  2
;  2368          2416  2       ! Deallocate the array of pointers to DDBs
;  2369          2417  2       !
;  2370          2418  2       SOR$$DEALLOCATE( %UPVAL * .CTX[COM_NUM_FILES], CTX[COM_INP_ARRAY]);
;  2371          2419  2
;  2372          2420  2
;  2373          2421  2       ! Deallocate all the DDBs
;  2374          2422  2       !
;  2375        L 2423  2       %IF NOT HOSTILE
;  2376          2424  2       %THEN
;  2377          2425  2           FREE_DDBS(CTX[COM_INP_DDB], SOR$_SHR_CLOSEIN,  TRUE);
;  2378          2426  2           FREE_DDBS(CTX[COM_OUT_DDB], SOR$_SHR_CLOSEOUT, TRUE);
;  2379          2427  2           FREE_DDBS(CTX[COM_SPC_DDB], SOR$_SHR_CLOSEIN,  FALSE);
;  2380          2428  2       %FI
;  2381          2429  2
;  2382          2430  2
;  2383          2431  2       ! Deallocate space for the record header buffer (VFC area)
;  2384          2432  2       !
;  2385          2433  2       SOR$$DEALLOCATE(.CTX[COM_MAXVFC], CTX[COM_RHB_INP]);
;  2386          2434  2       SOR$$DEALLOCATE(.CTX[COM_MAXVFC], CTX[COM_RHB_OUT]);
;  2387          2435  2
;  2388          2436  2
```

```
2      ! Deallocate the "pass files" information
2      !
2      SOR$$DEALLOCATE(%UPVAL+8*%UPVAL, CTX[COM_PASS_FILES]);
2
2
2      ! Deallocate the context area
2      !
2      CTX__[0] = .CTX__[0] AND NOT %B'1';          ! No longer active
2      SOR$$DEALLOCATE(CTX_K_SIZE*%UPVAL, CTX__[0]);
2
2
2      ! Dump the amount of unfreed space if compiled with debugging
2      !
L 2    %IF %SWITCHES(DEBUG)
U 2    %THEN
U 2        BEGIN
U 2        EXTERNAL ROUTINE LIB$STAT_VM: ADDRESSING_MODE(GENERAL);
U 2        EXTERNAL ROUTINE SOR$$OUTPUT;
U 2        MACRO
U 2            DESC_(A) = UPLIT(%CHARCOUNT(A), UPLIT BYTE(A)) %;
U 2        LOCAL GETS, FREES, UNFREED;
U 2        LIB$STAT_VM(%REF(1), GETS);
U 2        LIB$STAT_VM(%REF(2), FREES);
U 2        LIB$STAT_VM(%REF(3), UNFREED);
U 2        SOR$$OUTPUT(DESC_(
U 2            '!XL unfreed bytes = !UL pages, !UL bytes (!UL GETs, !UL FREEs)'),
U 2                .UNFREED, .UNFREED/COM_K_BPERPAGE, .UNFREED MOD COM_K_BPERPAGE,
U 2                .GETS, .FREES);
U 2        END;
2      %FI
2
2      RETURN SS$_NORMAL;
1      END;


                                                .PSECT  SOR$RO_CODE_______________________3,NOWRT,  SHR,  PIC,

                                00000 END_1:    .BLKB   0

                                                .PSECT  SOR$RO_CODE_______________________1,NOWRT,  SHR,  PIC,

                                00000 END_0:    .BLKB   0


                                                .PSECT  SOR$RO_CODE,NOWRT,  SHR,  PIC,2

                          087C 00000            .ENTRY  SOR$END_SORT, Save R2,R3,R4,R5,R6,R11        ; 2323
        56      F462   CF   9E 00002            MOVAB   SOR$$DEALLOCATE, R6                          ;
                       7E   D4 00007            CLRL    CTX__                                        ; 2367
        6D      00C4   CF   DE 00009            MOVAL   8$, (FP)                                     ;
                       6C   95 0000E            TSTB    (AP)                                         ; 2377
                       06   13 00010            BEQL    1$                                           ;
        53        04   AC   D0 00012            MOVL    CONTEXT, P__                                 ;
                       07   12 00016            BNEQ    2$                                           ;
        53 00000000'   EF   9E 00018 1$:        MOVAB   SOR_CONTEXT, P__                             ;
```

```
                  6E          53  D0 0001F 2$:   MOVL    P__, CTX__                  ;
      05          63          00  E3 00022      BBCS    #0, (P__), 3$               ;
                        0B5C  C6  16 00026      JSB     AST_END_SORT                ;
                                  04 0002A      RET                                 ;
      5B          63          01  CB 0002B 3$:   BICL3   #1, (P__), CTX              ;
                              06  12 0002F      BNEQ    4$                          ;
                          00  BE  D4 00031      CLRL    @CTX__                      ;
                            0096  31 00034      BRW     7$                          ;
            5C    AB          10  88 00037 4$:   BISB2   #16, 92(CTX)                ;
          0130    CB          01  D0 0003B      MOVL    #1, 304(CTX)                ;
                  52 00000000' EF 9E 00040      MOVAB   END_0, P                    ; 2391
                  53 00000000' EF 9E 00047      MOVAB   END_1, Q                    ; 2392
                  53          52  D1 0004E 5$:   CMPL    P, Q                        ; 2393
                              09  1E 00051      BGEQU   6$                          ;
      50          82          52  C1 00053      ADDL3   P, (P)+, RTN                ; 2396
                  60          00  FB 00057      CALLS   #0, (RTN)                   ; 2397
                              F2  11 0005A      BRB     5$                          ; 2393
                        00A4  CB  9F 0005C 6$:   PUSHAB  164(CTX)                    ; 2418
                  50      59  AB  9A 00060      MOVZBL  89(CTX), R0                 ;
      7E          50          02  78 00064      ASHL    #2, R0, -(SP)               ;
                  66          02  FB 00068      CALLS   #2, SOR$$DEALLOCATE         ;
                              01  DD 0006B      PUSHL   #1                          ; 2425
                  001C1052    8F  DD 0006D      PUSHL   #1839186                    ;
                        009C  CB  9F 00073      PUSHAB  156(CTX)                    ;
          0ACC    C6          03  FB 00077      CALLS   #3, FREE_DDBS               ;
                              01  DD 0007C      PUSHL   #1                          ; 2426
                  001C105A    8F  DD 0007E      PUSHL   #1839194                    ;
                        0098  CB  9F 00084      PUSHAB  152(CTX)                    ;
          0ACC    C6          03  FB 00088      CALLS   #3, FREE_DDBS               ;
                              7E  D4 0008D      CLRL    -(SP)                       ; 2427
                  001C1052    8F  DD 0008F      PUSHL   #1839186                    ;
                        00AC  CB  9F 00095      PUSHAB  172(CTX)                    ;
          0ACC    C6          03  FB 00099      CALLS   #3, FREE_DDBS               ;
                        008C  CB  9F 0009E      PUSHAB  140(CTX)                    ; 2433
                  7E    0082  CB  9A 000A2      MOVZBL  130(CTX), -(SP)             ;
                  66          02  FB 000A7      CALLS   #2, SOR$$DEALLOCATE         ;
                        0090  CB  9F 000AA      PUSHAB  144(CTX)                    ; 2434
                  7E    0082  CB  9A 000AE      MOVZBL  130(CTX), -(SP)             ;
                  66          02  FB 000B3      CALLS   #2, SOR$$DEALLOCATE         ;
                        0094  CB  9F 000B6      PUSHAB  148(CTX)                    ; 2439
                              24  DD 000BA      PUSHL   #36                         ;
                  66          02  FB 000BC      CALLS   #2, SOR$$DEALLOCATE         ;
            00    BE          01  8A 000BF      BICB2   #1, @CTX__                  ; 2444
                              6E  DD 000C3      PUSHL   CTX__                       ; 2445
                  7E    0150  8F  3C 000C5      MOVZWL  #336, -(SP)                 ;
                  66          02  FB 000CA      CALLS   #2, SOR$$DEALLOCATE         ;
                  50          01  D0 000CD 7$:   MOVL    #1, R0                      ; 2468
                                  04 000D0      RET                                 ; 2469
                                0000 000D1 8$:   .WORD   Save nothing                ; 2367
                  50      08  AC  D0 000D3      MOVL    8(AP), R0                   ;
                  50      04  A0  D0 000D7      MOVL    4(R0), R0                   ;
                          FC  A0  9F 000DB      PUSHAB  CTX__                       ;
                              01  DD 000DE      PUSHL   #1                          ;
                              5E  DD 000E0      PUSHL   SP                          ;
                  7E      04  AC  7D 000E2      MOVQ    4(AP), -(SP)                ;
          F1F3    CF          03  FB 000E6      CALLS   #3, COND_HAND               ;
                                  04 000EB      RET                                 ;
```

```
; Routine Size:  236 bytes,    Routine Base:  SOR$RO_CODE + 0D40
```

```
;  2423          2470  1 GLOBAL ROUTINE SOR$BEGIN_MERGE
;  2424          2471  1         (
;  2425          2472  1         KEY_BUFFER:     REF VECTOR[,WORD],      ! Key description buffer
;  2426          2473  1         LRL:            REF VECTOR[,WORD],      ! Longest record length
;  2427          2474  1         OPTIONS:        REF BLOCK[1],           ! Options
;  2428          2475  1         MERGE_ORDER:    REF VECTOR[,BYTE],      ! Order of the merge
;  2429          2476  1         USER_COMPARE,                           ! User comparison routine
;  2430          2477  1         USER_EQUAL,                             ! User equal-key routine
;  2431          2478  1         USER_INPUT,                             ! User input routine
;  2432          2479  1         CONTEXT:        REF VECTOR[1,LONG]      ! Addr of context longword
;  2433          2480  1         ) =
;  2434          2481  1 !++
;  2435          2482  1 !
;  2436          2483  1 ! FUNCTIONAL DESCRIPTION:
;  2437          2484  1 !
;  2438          2485  1 !       This routine initializes and does a merge.
;  2439          2486  1 !
;  2440          2487  1 ! FORMAL PARAMETERS:
;  2441          2488  1 !
;  2442          2489  1 !       MERGE_ORDER.rab         Order of the merge
;  2443          2490  1 !       KEY_BUFFER.raw          Key buffer address
;  2444          2491  1 !       LRL.rwu.r               Longest record length
;  2445          2492  1 !       OPTIONS.rlu.r           Option bits
;  2446          2493  1 !       USER_COMPARE,           User-written comparison routine
;  2447          2494  1 !       USER_INPUT              User-written input routine
;  2448          2495  1 !       USER_EQUAL,             User-written equal-key routine
;  2449          2496  1 !       CONTEXT.ra.r            Longword pointing to work area
;  2450          2497  1 !
;  2451          2498  1 !       All parameters are optional.
;  2452          2499  1 !
;  2453          2500  1 ! IMPLICIT INPUTS:
;  2454          2501  1 !
;  2455          2502  1 !       NONE
;  2456          2503  1 !
;  2457          2504  1 ! IMPLICIT OUTPUTS:
;  2458          2505  1 !
;  2459          2506  1 !       NONE
;  2460          2507  1 !
;  2461          2508  1 ! ROUTINE VALUE:
;  2462          2509  1 !
;  2463          2510  1 !       Status code.
;  2464          2511  1 !
;  2465          2512  1 ! SIDE EFFECTS:
;  2466          2513  1 !
;  2467          2514  1 !       The work files are defined (not necessarily created), the working set
;  2468          2515  1 !       is extended and the virtual memory is extended.
;  2469          2516  1 !
;  2470          2517  1 !--
;  2471          2518  2     BEGIN
;  2472          2519  2
;  2473          2520  2     ! Process the parameters
;  2474          2521  2     !
;  2475          2522  2     BEGIN_SORT_MERGE(MERGE)
;  2476          2523  2
;  2477          2524  2
;  2478          2525  2     ! Process the USER_INPUT parameter
;  2479          2526  2     !
```

```
3       IF NOT NULL_(USER_INPUT)
2       THEN
2           CTX[COM_MRG_INPUT] = .USER_INPUT;
2
2       ! If no files were passed, the user must specify an input routine
2       !
2       IF .CTX[COM_NUM_FILES] EQL 0 AND .CTX[COM_MRG_INPUT] EQL 0
2       THEN
2           RETURN__(SOR$_MISS_PARAM);
2
2       ! The merge will use TREE_EXTRACT to get records back from the streams.
2       ! Set the number of runs to a non-zero value so that TREE_EXTRACT realizes
2       ! it should not use the replacement selection tree.
2       !
2       CTX[COM_RUNS] = .CTX[COM_MRG_ORDER];
2
2       ! Link all the files to the VFC area
2       !
3       BEGIN
3       LOCAL
3           DDB:    REF DDB_BLOCK;
3       DDB = .CTX[COM_INP_DDB];                    ! Point to first DDB
3       DECR I FROM .CTX[COM_NUM_FILES]-1 TO 0 DO
4           BEGIN
4           DDB[DDB_RAB+RAB$L_RHB] = .CTX[COM_RHB_INP];
4           DDB = .DDB[DDB_NEXT];
3           END;
3       IF (DDB = .CTX[COM_OUT_DDB]) NEQ 0
3       THEN
3           DDB[DDB_RAB+RAB$L_RHB] = .CTX[COM_RHB_OUT];
2       END;
2
2       ! Call another routine to merge the sources into the output file.
2       ! If the record interface is used on output, this routine does nothing.
2       !
L 2     %IF NOT HOSTILE
2       %THEN
2           MRG_OUTPUT();
2       %FI
2
2       RETURN__(SS$_NORMAL);
1       END;
```

```
                        087C 00000        .ENTRY  SOR$BEGIN_MERGE, Save R2,R3,R4,R5,R6,R11    ; 2470
56      F2D5    CF   9E 00002        MOVAB   SOR$$ERROR, R6                              ;
5E              10   C2 00007        SUBL2   #16, SP                                     ;
          0C    AE   D4 0000A        CLRL    CTX__                                        ; 2518
6D      03AA    CF   DE 0000D        MOVAL   58$, (FP)                                   ;
08              6C   91 00012        CMPB    (AP), #8                                    ; 2522
```

```
                                    06  1F 00015          BLSSU   1$                                                                ;
                          53     20   AC  D0 00017          MOVL    CONTEXT, P__                                                      ;
                                      07  12 0001B          BNEQ    2$                                                                ;
                          53 00000000' EF  9E 0001D 1$:      MOVAB   SOR_CONTEXT, P__                                                  ;
                       0C AE          53  D0 00024 2$:      MOVL    P__, CTX                                                          ;
                    31 63             00  E2 00028          BBSS    #0, (P__), 7$                                                     ;
                    5B 63             01  CB 0002C          BICL3   #1, (P__), CTX                                                    ;
                                      03  12 00030          BNEQ    3$                                                                ;
                             13       A6  16 00032          JSB     INI_CTX                                                           ;
                          51 5C        AB  9E 00035 3$:      MOVAB   92(CTX), R1                                                       ;
                    1C    61           04  E2 00039          BBSS    #4, (R1), 6$                                                      ;
                       0130 CB           01  D0 0003D          MOVL    #1, 304(CTX)                                                      ;
                          03             6C  91 00042          CMPB    (AP), #3                                                          ;
                                         0A  1F 00045          BLSSU   4$                                                                ;
                             0C       AC  D5 00047          TSTL    12(AP)                                                            ;
                                      05  13 0004A          BEQL    4$                                                                ;
                    04    0C BC           03  E0 0004C          BBS     #3, @OPTIONS, 5$                                                  ;
                       5B AB              04  88 00051 4$:      BISB2   #4, 91(CTX)                                                       ;
                    0C    61              00  E3 00055 5$:      BBCS    #0, (R1), 8$                                                      ;
                       0C BE              01  8A 00059 6$:      BICB2   #1, @CTX__                                                        ;
                          50 001C802C     8F  D0 0005D 7$:      MOVL    #1867820, R0                                                      ;
                                             04 00064          RET                                                                       ;
                                0098     CB  D5 00065 8$:      TSTL    152(CTX)                                                          ;
                                         03  12 00069          BNEQ    9$                                                                ;
                          61             04  88 0006B          BISB2   #4, (R1)                                                          ;
                          61             08  88 0006E 9$:      BISB2   #8, (R1)                                                          ;
                          03             6C  91 00071          CMPB    (AP), #3                                                          ;
                                         5D  1F 00074          BLSSU   17$                                                               ;
                                   0C    AC  D5 00076          TSTL    12(AP)                                                            ;
                                         58  13 00079          BEQL    17$                                                               ;
                             50       0C   AC  D0 0007B          MOVL    OPTIONS, R0                                                       ;
                FFFFFDA0     8F            60  D3 0007F          BITL    (R0), #-608                                                       ;
                                           0C  13 00086          BEQL    10$                                                               ;
                       0C    BE            01  8A 00088          BICB2   #1, @CTX__                                                        ;
                             50 001C814C   8F  D0 0008C          MOVL    #1868108, R0                                                      ;
                                           04  00093          RET                                                                       ;
                    04       60            04  E1 00094 10$:     BBC     #4, (R0), 11$                                                     ;
                       5B    AB            02  88 00098          BISB2   #2, 91(CTX)                                                       ;
                             04            60  E9 0009C 11$:     BLBC    (R0), 12$                                                         ;
                       5B    AB            01  88 0009F          BISB2   #1, 91(CTX)                                                       ;
                    04       60            06  E1 000A3 12$:     BBC     #6, (R0), 13$                                                     ;
                       5B    AB            20  88 000A7          BISB2   #32, 91(CTX)                                                      ;
                    04       60            09  E1 000AB 13$:     BBC     #9, (R0), 14$                                                     ;
                       5B    AB            10  88 000AF          BISB2   #16, 91(CTX)                                                      ;
                    08       60            01  E0 000B3 14$:     BBS     #1, (R0), 15$                                                     ;
                    18       60            02  E1 000B7          BBC     #2, (R0), 17$                                                     ;
                    10       60            01  E1 000BB          BBC     #1, (R0), 16$                                                     ;
                    0C       60            02  E1 000BF 15$:     BBC     #2, (R0), 16$                                                     ;
                       0C    BE            01  8A 000C3          BICB2   #1, @CTX__                                                        ;
                             50 001C8134   8F  D0 000C7          MOVL    #1868084, R0                                                      ;
                                           04  000CE          RET                                                                       ;
                       79    AB            04  88 000CF 16$:     BISB2   #4, 121(CTX)                                                      ;
                             61       80   8F  88 000D3 17$:     BISB2   #128, (R1)                                                        ;
                             52       58   AB  9E 000D7          MOVAB   88(CTX), R2                                                       ;
                             62            01  90 000DB          MOVB    #1, (R2)                                                          ;
                       01    A1       01   A2  90 000DE          MOVB    1(R2), 1(R1)                                                      ;
                                           28  12 000E3          BNEQ    20$                                                               ;
```

```
                04          6C  91 000E5          CMPB    (AP), #4                    ;
                            0A  1F 000E8          BLSSU   18$                         ;
                      10    AC  D5 000EA          TSTL    16(AP)                      ;
                            05  13 000ED          BEQL    18$                         ;
          01    A1    10    BC  90 000EF          MOVB    @MERGE_ORDER, 1(R1)         ;
                      01    A1  95 000F4 18$:     TSTB    1(R1)                       ;
                            06  13 000F7          BEQL    19$                         ;
                0A    01    A1  91 000F9          CMPB    1(R1), #10                  ;
                            0E  1B 000FD          BLEQU   20$                         ;
          0C    BE          01  8A 000FF 19$:     BICB2   #1, @CTX                    ;
                001C80BA    8F  DD 00103          PUSHL   #1867962                    ;
                66          01  FB 00109          CALLS   #1, SOR$$ERROR              ;
                            04  0010C             RET                                 ;
                02          6C  91 0010D 20$:     CMPB    (AP), #2                    ;
                            0B  1F 00110          BLSSU   21$                         ;
                      08    AC  D5 00112          TSTL    8(AP)                       ;
                            06  13 00115          BEQL    21$                         ;
        0084    CB    08    BC  B0 00117          MOVW    @LRL, 132(CTX)              ;
                06          6C  91 0011D 21$:     CMPB    (AP), #6                    ;
                            19  1F 00120          BLSSU   22$                         ;
                      18    AC  D5 00122          TSTL    24(AP)                      ;
                            14  13 00125          BEQL    22$                         ;
          04    AB    18    AC  D0 00127          MOVL    USER_EQUAL, 4(CTX)          ;
  0F            62          1D  E5 0012C          BBCC    #29, (R2), 23$              ;
                001C8104    8F  DD 00130          PUSHL   #1868036                    ;
                66          01  FB 00136          CALLS   #1, SOR$$ERROR              ;
                            04  11 00139          BRB     23$                         ;
  0D            62          1D  E1 0013B 22$:     BBC     #29, (R2), 24$              ;
  09            62          18  E5 0013F 23$:     BBCC    #24, (R2), 24$              ;
                001C8154    8F  DD 00143          PUSHL   #1868116                    ;
                66          01  FB 00149          CALLS   #1, SOR$$ERROR              ;
                05          6C  91 0014C 24$:     CMPB    (AP), #5                    ;
                            05  1F 0014F          BLSSU   25$                         ;
                      14    AC  D5 00151          TSTL    20(AP)                      ;
                            09  12 00154          BNEQ    26$                         ;
                            6C  95 00156 25$:     TSTB    (AP)                        ;
                            09  13 00158          BEQL    27$                         ;
                      04    AC  D5 0015A          TSTL    4(AP)                       ;
                            04  13 0015D          BEQL    27$                         ;
          79    AB          02  88 0015F 26$:     BISB2   #2, 121(CTX)                ;
                    00AC    CB  D5 00163 27$:     TSTL    172(CTX)                    ;
                            06  12 00167          BNEQ    28$                         ;
                    00F4    CB  B5 00169          TSTW    244(CTX)                    ;
                            07  13 0016D          BEQL    29$                         ;
    00000000G   00          00  FB 0016F 28$:     CALLS   #0, SOR$$SPEC_FILE          ;
                03          6C  91 00176 29$:     CMPB    (AP), #3                    ;
                            03  1E 00179          BGEQU   31$                         ;
                          00F3  31 0017B 30$:     BRW     41$                         ;
                      0C    AC  D5 0017E 31$:     TSTL    12(AP)                      ;
                            F8  13 00181          BEQL    30$                         ;
  0B        0C    BC          01  E0 00183          BBS     #1, @OPTIONS, 32$           ;
  06        0C    BC          02  E0 00188          BBS     #2, @OPTIONS, 32$           ;
                    011C    CB  D5 0018D          TSTL    284(CTX)                    ;
                            E8  13 00191          BEQL    30$                         ;
                52    0124  CB  9E 00193 32$:     MOVAB   292(CTX), R2                ;
                            62  D5 00198          TSTL    (R2)                        ;
                            2D  12 0019A          BNEQ    34$                         ;
```

```
                        62      0600   8F  3C 0019C          MOVZWL  #1536, (R2)                          ;
                                0128   CB  9F 001A1          PUSHAB  296(CTX)                             ;
                                       52  DD 001A5          PUSHL   R2                                   ;
            00000000G   00             02  FB 001A7          CALLS   #2, LIB$GET_VM                       ;
                        53             50  D0 001AE          MOVL    R0, STATUS                           ;
                        0D             53  E8 001B1          BLBS    STATUS, 33$                          ;
                                       53  DD 001B4          PUSHL   STATUS                               ;
                                       7E  D4 001B6          CLRL    -(SP)                                ;
                        001C11B4       8F  DD 001B8          PUSHL   #1839540                             ;
                        66             03  FB 001BE          CALLS   #3, SOR$$ERROR                       ;
    012C   CB     0128  CB             62  C1 001C1 33$:     ADDL3   (R2), 296(CTX), 300(CTX)             ;
      04   AE     012C  CB      0128   CB  C3 001C9 34$:     SUBL3   296(CTX), 300(CTX), COLL_DESC        ;
                   08   AE      0128   CB  D0 001D2          MOVL    296(CTX), COLL_DESC+4                ;
                                  04   AE  9F 001D8          PUSHAB  COLL_DESC                            ;
            00000000G   00             01  FB 001DB          CALLS   #1, COLL$INIT                        ;
                        53             50  D0 001E2          MOVL    R0, STATUS                           ;
                        78             53  E9 001E5          BLBC    STATUS, 39$                          ;
         0C       0C   BC             02  E1 001E8          BBC     #2, @OPTIONS, 35$                    ;
                                  04   AE  9F 001ED          PUSHAB  COLL_DESC                            ;
            00000000G   00             01  FB 001F0          CALLS   #1, SOR$$DECM                        ;
                                       1F  11 001F7          BRB     37$                                  ;
         0C       0C   BC             01  E1 001F9 35$:     BBC     #1, @OPTIONS, 36$                    ;
                                  04   AE  9F 001FE          PUSHAB  COLL_DESC                            ;
            00000000G   00             01  FB 00201          CALLS   #1, SOR$$EBCDIC                      ;
                                       0E  11 00208          BRB     37$                                  ;
                                011C   CB  DD 0020A 36$:     PUSHL   284(CTX)                             ;
                                  08   AE  9F 0020E          PUSHAB  COLL_DESC                            ;
            00000000G   00             02  FB 00211          CALLS   #2, COLL$BASE                        ;
                        53             50  D0 00218 37$:     MOVL    R0, STATUS                           ;
                        42             53  E9 0021B          BLBC    STATUS, 39$                          ;
                        6E             01  B0 0021E          MOVW    #1, PAD_CHAR                         ;
                   02   AE      0101   CB  90 00221          MOVB    257(CTX), PAD_CHAR+2                 ;
                                       5E  DD 00227          PUSHL   SP                                   ;
                                  08   AE  9F 00229          PUSHAB  COLL_DESC                            ;
            00000000G   00             02  FB 0022C          CALLS   #2, COLL$PAD                         ;
                        53             50  D0 00233          MOVL    R0, STATUS                           ;
                        27             53  E9 00236          BLBC    STATUS, 39$                          ;
                        12      0080   CB  E9 00239          BLBC    128(CTX), 38$                        ;
                                       01  DD 0023E          PUSHL   #1                                   ;
                                  08   AE  9F 00240          PUSHAB  COLL_DESC                            ;
            00000000G   00             02  FB 00243          CALLS   #2, COLL$TIE_BREAK                   ;
                        53             50  D0 0024A          MOVL    R0, STATUS                           ;
                        10             53  E9 0024D          BLBC    STATUS, 39$                          ;
                                  04   AE  9F 00250 38$:     PUSHAB  COLL_DESC                            ;
                                  08   AE  9F 00253          PUSHAB  COLL_DESC                            ;
            00000000G   00             02  FB 00256          CALLS   #2, COLL$RESULT                      ;
                        53             50  D0 0025D          MOVL    R0, STATUS                           ;
                        03             53  E8 00260 39$:     BLBS    STATUS, 40$                          ;
                                     0091  31 00263          BRW     47$                                  ;
                0128   CB        04   AE  C0 00266 40$:     ADDL2   COLL_DESC, 296(CTX)                  ;
                  68   AB        08   AE  D0 0026C          MOVL    COLL_DESC+4, 104(CTX)                ;
                                  59   AB  95 00271 41$:     TSTB    89(CTX)                              ;
                                       29  12 00274          BNEQ    43$                                  ;
                0080   CB             02  88 00276          BISB2   #2, 128(CTX)                         ;
                        01        58   AB  91 0027B          CMPB    88(CTX), #1                          ;
                                       0C  13 0027F          BEQL    42$                                  ;
                  0C   BE             01  8A 00281          BICB2   #1, @CTX__                           ;
```

```
                50 001C806C   8F   D0 00285          MOVL    #1867884, R0
                                04 0028C          RET
                      0084    CB   B5 0028D 42$:    TSTW    132(CTX)
                              0C   12 00291          BNEQ    43$
              0C      BE              01 8A 00293   BICB2   #1, @CTX
                50 001C8074   8F   D0 00297          MOVL    #1867892, R0
                                04 0029E          RET
                54 00000000G  00   9E 0029F 43$:    MOVAB   SOR$$KEY_SUB, DOKEY_RTN
                              52   D4 002A6          CLRL    DOKEY_PRM
                05            6C   91 002A8          CMPB    (AP), #5
                              1D   1F 002AB          BLSSU   44$
                         14   AC   D5 002AD          TSTL    20(AP)
                              18   13 002B0          BEQL    44$
                6B       14   AC   D0 002B2          MOVL    USER_COMPARE, (CTX)
                              6C   95 002B6          TSTB    (AP)
                              2C   13 002B8          BEQL    46$
                         04   AC   D5 002BA          TSTL    4(AP)
                              27   13 002BD          BEQL    46$
                   001C8134   8F   DD 002BF          PUSHL   #1868084
                66            01   FB 002C5          CALLS   #1, SOR$$ERROR
                              1C   11 002C8          BRB     46$
                              6C   95 002CA 44$:     TSTB    (AP)
                              0B   13 002CC          BEQL    45$
                         04   AC   D5 002CE          TSTL    4(AP)
                              06   13 002D1          BEQL    45$
                52       04   AC   D0 002D3          MOVL    KEY_BUFFER, DOKEY_PRM
                              0D   11 002D7          BRB     46$
                      00FC    CB   95 002D9 45$:     TSTB    252(CTX)
                              07   13 002DD          BEQL    46$
                54 00000000G  00   9E 002DF          MOVAB   SOR$$SPEC_KEY_SUB, DOKEY_RTN
                              52   DD 002E6 46$:     PUSHL   DOKEY_PRM
                              54   DD 002E8          PUSHL   DOKEY_RTN
          00000000G   00            02 FB 002EA     CALLS   #2, SOR$$OPEN
                      53            50 D0 002F1     MOVL    R0, STATUS
                      08            53 E8 002F4     BLBS    STATUS, 48$
              0C      BE            01 8A 002F7 47$: BICB2  #1, @CTX
                      50            53 D0 002FB     MOVL    STATUS, R0
                                    04 002FE        RET
                      0094    CB    9F 002FF 48$:   PUSHAB  148(CTX)
                              24    DD 00303        PUSHL   #36
                00A1  C6            02 FB 00305     CALLS   #2, SOR$$DEALLOCATE
                      50      0082  CB 9A 0030A     MOVZBL  130(CTX), R0
                                    19 13 0030F     BEQL    49$
                              50    DD 00311        PUSHL   R0
                   72 A6            01 FB 00313     CALLS   #1, SOR$$ALLOCATE
                008C  CB            50 D0 00317     MOVL    R0, 140(CTX)
                      7E      0082  CB 9A 0031C     MOVZBL  130(CTX), -(SP)
                   72 A6            01 FB 00321     CALLS   #1, SOR$$ALLOCATE
                0090  CB            50 D0 00325     MOVL    R0, 144(CTX)
                      01      58    AB 91 0032A 49$: CMPB   88(CTX), #1
                                    2A 13 0032E     BEQL    52$
                      50      59    AB 9A 00330     MOVZBL  89(CTX), R0
          7E          50            02 78 00334     ASHL    #2, R0, -(SP)
                   72 A6            01 FB 00338     CALLS   #1, SOR$$ALLOCATE
                00A4  CB            50 D0 0033C     MOVL    V, 164(CTX)
                      52      009C  CB D0 00341     MOVL    156(CTX), P
                      53        59  AB 9A 00346     MOVZBL  89(CTX), I
```

```
                                0B  11 0034A          BRB     51$                                  ;
          51          58        A2  9A 0034C 50$:     MOVZBL  88(P), R1                            ;
        6041                    52  D0 00350          MOVL    P, (V)[R1]                           ;
          52                    62  D0 00354          MOVL    (P), P                               ;
          F2                    53  F4 00357 51$:     SOBGEQ  I, 50$                               ;
          07                    6C  91 0035A 52$:     CMPB    (AP), #7                             ; 2527
                                0A  1F 0035D          BLSSU   53$                                  ;
                        1C      AC  D5 0035F          TSTL    28(AP)                               ;
                                05  13 00362          BEQL    53$                                  ;
      60    AB          1C      AC  D0 00364          MOVL    USER_INPUT, 96(CTX)                  ; 2529
                        59      AB  95 00369 53$:     TSTB    89(CTX)                              ; 2534
                                11  12 0036C          BNEQ    54$                                  ;
                        60      AB  D5 0036E          TSTL    96(CTX)                              ;
                                0C  12 00371          BNEQ    54$                                  ;
      0C    BE                  01  8A 00373          BICB2   #1, @CTX__                           ; 2536
          50 001C80E4           8F  D0 00377          MOVL    #1868004, R0                         ;
                                    04 0037E          RET                                          ;
      7A    AB          5D      AB  9B 0037F 54$:     MOVZBW  93(CTX), 122(CTX)                    ; 2543
          50        009C        CB  D0 00384          MOVL    156(CTX), DDB                        ; 2551
          51          59        AB  9A 00389          MOVZBL  89(CTX), I                           ; 2554
                                09  11 0038D          BRB     56$                                  ;
      40    A0        008C      CB  D0 0038F 55$:     MOVL    140(CTX), 64(DDB)                    ;
          50                    60  D0 00395          MOVL    (DDB), DDB                           ; 2555
          F4                    51  F4 00398 56$:     SOBGEQ  I, 55$                               ; 2552
          50        0098        CB  D0 0039B          MOVL    152(CTX), DDB                        ; 2557
                                06  13 003A0          BEQL    57$                                  ;
      40    A0        0090      CB  D0 003A2          MOVL    144(CTX), 64(DDB)                    ; 2559
    08A6    C6                  00  FB 003A8 57$:     CALLS   #0, MRG_OUTPUT                       ; 2568
      0C    BE                  01  8A 003AD          BICB2   #1, @CTX__                           ; 2571
      5C    AB                  10  8A 003B1          BICB2   #16, 92(CTX)                         ;
          50        0130        CB  D0 003B5          MOVL    304(CTX), R0                         ;
                                    04 003BA          RET                                          ; 2572
                                  0000 003BB 58$:     .WORD   Save nothing                         ; 2518
          50          08        AC  D0 003BD          MOVL    8(AP), R0                            ;
          50          04        A0  D0 003C1          MOVL    4(R0), R0                            ;
                        FC        A0  9F 003C5          PUSHAB  CTX__                                ;
                                01  DD 003C8          PUSHL   #1                                   ;
                                5E  DD 003CA          PUSHL   SP                                   ;
          7E          04        AC  7D 003CC          MOVQ    4(AP), -(SP)                         ;
    EE1D    CF                  03  FB 003D0          CALLS   #3, COND_HAND                        ;
                                    04 003D5          RET                                          ;

; Routine Size:  982 bytes,    Routine Base:  SOR$RO_CODE + 0E2C
```

```
;  2527          2573  1 %IF NOT HOSTILE %THEN
;  2528          2574  1 GLOBAL ROUTINE SOR$SPEC_FILE
;  2529          2575  1         (
;  2530          2576  1         SPC_DESC:       REF BLOCK[,BYTE],       ! Filename descriptor
;  2531          2577  1         TXT_DESC:       REF BLOCK[,BYTE],       ! Text descriptor
;  2532          2578  1         CONTEXT:        REF VECTOR[1,LONG],     ! Addr of context longword
;  2533          2579  1         COLL_BLOCK:     REF BLOCK[,BYTE]        ! Collate sequence description
;  2534          2580  1         ) =
;  2535          2581  1 !++
;  2536          2582  1 !
;  2537          2583  1 ! FUNCTIONAL DESCRIPTION:
;  2538          2584  1 !
;  2539          2585  1 !       This routine processes the specification file.
;  2540          2586  1 !
;  2541          2587  1 ! FORMAL PARAMETERS:
;  2542          2588  1 !
;  2543          2589  1 !       SPC_DESC.ra.d   Specification file descriptor
;  2544          2590  1 !       TXT_DESC.ra.d   Specificatoin text descriptor
;  2545          2591  1 !       CONTEXT.ra.r    Longword pointing to work area
;  2546          2592  1 !       COLL_BLOCK.ra.r Collating sequence description
;  2547          2593  1 !
;  2548          2594  1 !       All parameters are optional.
;  2549          2595  1 !
;  2550          2596  1 ! IMPLICIT INPUTS:
;  2551          2597  1 !
;  2552          2598  1 !       NONE
;  2553          2599  1 !
;  2554          2600  1 ! IMPLICIT OUTPUTS:
;  2555          2601  1 !
;  2556          2602  1 !       NONE
;  2557          2603  1 !
;  2558          2604  1 ! ROUTINE VALUE:
;  2559          2605  1 !
;  2560          2606  1 !       Status code.
;  2561          2607  1 !
;  2562          2608  1 ! SIDE EFFECTS:
;  2563          2609  1 !
;  2564          2610  1 !       NONE
;  2565          2611  1 !
;  2566          2612  1 !--
;  2567          2613  2     BEGIN
;  2568          2614  2     FIRSTPARAMETER_(SPC_DESC);  ! Required by PRESENT_ and NULL_ macros
;  2569          2615  2
;  2570          2616  2     LOCAL
;  2571          2617  2         STATUS;
;  2572          2618  2
;  2573          2619  2     CONTEXT_(CONTEXT);
;  2574          2620  2
;  2575          2621  2     IF .CTX[COM_FLO_NOINIT] THEN RETURN__(SOR$_SORT_ON);
;  2576          2622  2
;  2577          2623  2     ! Check for spec file name present
;  2578          2624  2     !
;  2579          2625  3     IF NOT NULL_(SPC_DESC)
;  2580          2626  2     THEN
;  2581          2627  3         BEGIN
;  2582          2628  3         LOCAL
;  2583          2629  3             DDB:        REF DDB_BLOCK,
```

```
;  2584          2630  3              P:          REF DDB_BLOCK;
;  2585          2631  3
;  2586          2632  3          ! Allocate DDB and link into list (at the end of the list)
;  2587          2633  3          !
;  2588          2634  3          DDB = SOR$$ALLOCATE(DDB_K_SIZE);
;  2589          2635  3          P = CTX[COM_SPC_DDB];
;  2590          2636  3          WHILE .P[DDB_NEXT] NEQ 0 DO P = .P[DDB_NEXT];
;  2591          2637  3          P[DDB_NEXT] = DDB[BASE_];                       ! Link DDB into list
;  2592          2638  3
;  2593          2639  3          ! Copy the input file name string to the DDB
;  2594          2640  3          !
;  2595          2641  3          SOR$$COPY_FILE_NAME(SPC_DESC[BASE_], DDB[DDB_NAME]);
;  2596          2642  2          END;
;  2597          2643  2
;  2598          2644  2
;  2599          2645  3      BEGIN
;  2600          2646  3      BIND
;  2601          2647  3          TXT = CTX[COM_SPC_TXT]: BLOCK[,BYTE];
;  2602          2648  3      TXT[DSC$B_CLASS] = DSC$K_CLASS_D;
;  2603          2649  4      IF NOT NULL_(TXT_DESC)
;  2604          2650  3      THEN
;  2605          2651  4          BEGIN
;  2606          2652  4          !
;  2607          2653  4          ! Append the text to the end of our dynamic string.
;  2608          2654  4          !
;  2609          2655  4          STATUS = STR$APPEND(TXT[BASE_], TXT_DESC[BASE_]);
;  2610          2656  4          IF NOT .STATUS THEN RETURN__(SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS));
;  2611          2657  3          END;
;  2612          2658  2      END;
;  2613          2659  2
;  2614          2660  2
;  2615          2661  3      BEGIN
;  2616          2662  4      IF NOT NULL_(COLL_BLOCK)
;  2617          2663  3      THEN
;  2618          2664  4          BEGIN
;  2619          2665  4          !
;  2620          2666  4          ! Save the pad character and pointer to the 256-byte table
;  2621          2667  4          !
;  2622          2668  4          ! Note that it is very tacky and inconsistent to NOT actually make
;  2623          2669  4          ! a copy of this table.  However, this interface is only known to
;  2624          2670  4          ! VAX COBOL, and the table they pass hangs around.
;  2625          2671  4          !
;  2626          2672  4          MACRO
;  2627          2673  4              HAS_(O,P,S,E) = (.COLL_BLOCK[COLL_W_LENGTH] GEQU O+S/%BPUNIT) %;
;  2628          2674  4          IF HAS_(COLL_B_PAD)  THEN CTX[COM_PAD]  = .COLL_BLOCK[COLL_B_PAD];
;  2629          2675  4          IF HAS_(COLL_A_PTAB) THEN CTX[COM_PTAB] = .COLL_BLOCK[COLL_A_PTAB];
;  2630          2676  3          END;
;  2631          2677  2      END;
;  2632          2678  2
;  2633          2679  2
;  2634          2680  2      RETURN__(SS$_NORMAL);
;  2635          2681  1      END;
```

```
                                    083C 00000          .ENTRY  SOR$SPEC_FILE, Save R2,R3,R4,R5,R11     ; 2574
                                 7E  D4 00002           CLRL    CTX                                     ; 2613
                          6D     CF  DE 00004           MOVAL   13$, (FP)
                          03     6C  91 00009           CMPB    (AP), #3                                ; 2619
                                 06  1F 0000C           BLSSU   1$
                          53     0C  AC D0 0000E        MOVL    CONTEXT, P
                                 07  12 00012           BNEQ    2$
                          53 00000000'  EF 9E 00014 1$:  MOVAB   SOR_CONTEXT, P
                          6E         53  D0 0001B 2$:   MOVL    P, CTX
                    1B    63             00  E2 0001E   BBSS    #0, (P), 5$
                    5B    63             01  CB 00022   BICL3   #1, (P), CTX
                                 03  12 00026           BNEQ    3$
                                EEED 30 00028           BSBW    INI_CTX
                    09  5C    AB             04  E2 0002B 3$:  BBSS    #4, 92(CTX), 4$
                  0130  CB             01  D0 00030     MOVL    #1, 304(CTX)
                        0C     5C    AB  E9 00035       BLBC    92(CTX), 6$                             ; 2621
                        00     BE        01  8A 00039 4$:  BICB2   #1, @CTX
                          50 001C802C  8F  D0 0003D 5$:   MOVL    #1867820, R0
                                       04  00044         RET
                                 6C  95 00045 6$:        TSTB    (AP)                                    ; 2625
                                 2C  13 00047           BEQL    9$
                             04  AC  D5 00049           TSTL    4(AP)
                                 27  13 0004C           BEQL    9$
                          7E     59  8F 9A 0004E        MOVZBL  #89, -(SP)                              ; 2634
                    EF20  CF             01  FB 00052   CALLS   #1, SOR$$ALLOCATE
                          51     00AC    CB 9E 00057     MOVAB   172(R11), P                             ; 2635
                                 61  D5 0005C 7$:        TSTL    (P)                                     ; 2636
                                 05  13 0005E           BEQL    8$
                          51         61  D0 00060       MOVL    (P), P
                                 F7  11 00063           BRB     7$
                          61         50  D0 00065 8$:   MOVL    DDB, (P)                                ; 2637
                             04  A0  9F 00068           PUSHAB  4(DDB)                                  ; 2641
                             04  AC  DD 0006B           PUSHL   SPC_DESC
                00000000G    00         02  FB 0006E     CALLS   #2, SOR$$COPY_FILE_NAME
                          50     00F4    CB 9E 00075 9$:  MOVAB   244(R11), R0                           ; 2647
                        03   A0             02  90 0007A   MOVB    #2, 3(R0)                           ; 2648
                          02             6C  91 0007E     CMPB    (AP), #2                                ; 2649
                                 28  1F 00081           BLSSU   10$
                             08  AC  D5 00083           TSTL    8(AP)
                                 23  13 00086           BEQL    10$
                             08  AC  DD 00088           PUSHL   TXT_DESC                                ; 2655
                                 50  DD 0008B           PUSHL   R0
                00000000G    00         02  FB 0008D     CALLS   #2, STR$APPEND
                                 14  50 E8 00094        BLBS    STATUS, 10$                             ; 2656
                        00   BE         01  8A 00097     BICB2   #1, @CTX
                                 50  DD 0009B           PUSHL   STATUS
                                 7E  D4 0009D           CLRL    -(SP)
                             001C11B4    8F  DD 0009F   PUSHL   #1839540
                    EE5B  CF             03  FB 000A5   CALLS   #3, SOR$$ERROR
                                       04  000AA         RET
                          04             6C  91 000AB 10$:  CMPB    (AP), #4                            ; 2662
                                 1F  1F 000AE           BLSSU   12$
                             10  AC  D5 000B0           TSTL    16(AP)
                                 1A  13 000B3           BEQL    12$
                          50     10  AC  D0 000B5       MOVL    COLL_BLOCK, R0                          ; 2674
                          04             60  B1 000B9     CMPW    (R0), #4
                                 06  1F 000BC           BLSSU   11$
```

```
                0101   CB      03   A0  90 000BE          MOVB    3(R0), 257(CTX)                  ;
                       08           60  B1 000C4 11$:     CMPW    (R0), #8                         ; 2675
                                    06  1F 000C7          BLSSU   12$                              ;
                011C   CB      04   A0  D0 000C9          MOVL    4(R0), 284(CTX)                  ;
                  00   BE           01  8A 000CF 12$:     BICB2   #1, @CTX                         ; 2680
                  5C   AB           10  8A 000D3          BICB2   #16, 92(CTX)                     ;
                       50    0130   CB  D0 000D7          MOVL    304(CTX), R0                     ;
                                        04 000DC          RET                                      ; 2681
                                      0000 000DD 13$:     .WORD   Save nothing                     ; 2613
                       50      08   AC  D0 000DF          MOVL    8(AP), R0                        ;
                       50      04   A0  D0 000E3          MOVL    4(R0), R0                        ;
                               FC   A0  9F 000E7          PUSHAB  CTX__                            ;
                                    01  DD 000EA          PUSHL   #1                               ;
                                    5E  DD 000EC          PUSHL   SP                               ;
                       7E      04   AC  7D 000EE          MOVQ    4(AP), -(SP)                     ;
                ED25   CF           03  FB 000F2          CALLS   #3, COND_HAND                    ;
                                        04 000F7          RET                                      ;

; Routine Size:  248 bytes,    Routine Base:  SOR$RO_CODE + 1202

; 2636          2682  1 %FI
```

```
;  2638     2683  1 GLOBAL ROUTINE SOR$STAT
;  2639     2684  1         (
;  2640     2685  1         CODE:           REF VECTOR[1,LONG],     ! Addr of code
;  2641     2686  1         RESULT:         REF VECTOR[1,LONG],     ! Addr to store result
;  2642     2687  1         CONTEXT:        REF VECTOR[1,LONG]      ! Addr of context longword
;  2643     2688  1         ) =
;  2644     2689  1 !++
;  2645     2690  1 !
;  2646     2691  1 ! FUNCTIONAL DESCRIPTION:
;  2647     2692  1 !
;  2648     2693  1 !       This routine returns a statistic about the sort/merge.
;  2649     2694  1 !
;  2650     2695  1 ! FORMAL PARAMETERS:
;  2651     2696  1 !
;  2652     2697  1 !       CODE.lr.r       Code for requested statistic
;  2653     2698  1 !       RESULT.lw.r     Result area
;  2654     2699  1 !       CONTEXT.ra.r    Longword pointing to work area
;  2655     2700  1 !
;  2656     2701  1 !       All parameters except CODE and RESULT are optional.
;  2657     2702  1 !
;  2658     2703  1 ! IMPLICIT INPUTS:
;  2659     2704  1 !
;  2660     2705  1 !       NONE
;  2661     2706  1 !
;  2662     2707  1 ! IMPLICIT OUTPUTS:
;  2663     2708  1 !
;  2664     2709  1 !       NONE
;  2665     2710  1 !
;  2666     2711  1 ! ROUTINE VALUE:
;  2667     2712  1 !
;  2668     2713  1 !       Status code.
;  2669     2714  1 !
;  2670     2715  1 ! SIDE EFFECTS:
;  2671     2716  1 !
;  2672     2717  1 !       NONE
;  2673     2718  1 !
;  2674     2719  1 !--
;  2675     2720  2     BEGIN
;  2676     2721  2     FIRSTPARAMETER_(CODE);      ! Required by PRESENT_ and NULL_ macros
;  2677     2722  2
;  2678     2723  2     UNDECLARE
;  2679     2724  2         %QUOTE COM_FLO_ABORT;   ! Don't mess with this bit!
;  2680     2725  2
;  2681     2726  2     CONTEXT_(CONTEXT, SOR$_SORT_ON);
;  2682     2727  2
;  2683     2728  2     IF NULL_(RESULT) OR NULL_(CODE) THEN RETURN SOR$_MISS_PARAM;
;  2684     2729  2
;  2685     2730  2     RESULT[0] =
;  2686     2731  3         (BIND
;  2687     2732  3         INP = CTX[COM_INP_DDB]: REF DDB_BLOCK,
;  2688     2733  3         OUT = CTX[COM_OUT_DDB]: REF DDB_BLOCK;
;  2689     2734  3         CASE .CODE[0] FROM 0 TO STAT_K_MAX_STAT OF
;  2690     2735  3         SET
;  2691     2736  3         [STAT_K_IDENT]:         UPLIT(%ASCIC IDENT);    ! Address of ASCIC string for version number
;  2692     2737  3         [STAT_K_REC_INP]:       .CTX[COM_INP_RECNUM];   ! Records Input
;  2693     2738  3         [STAT_K_REC_SOR]:                               ! Records Sorted
;  2694     2739  3             .CTX[COM_INP_RECNUM] - .CTX[COM_OMI_RECNUM];
```

```
; 2695        2740  3        [STAT_K_REC_OUT]:        .CTX[COM_OUT_RECNUM];    ! Records Output
; 2696        2741  3        [STAT_K_LRL_INP]:        .CTX[COM_LRL];           ! LRL for Input
; 2697        2742  3        [STAT_K_LRL_INT]:        .CTX[COM_LRL_INT];       ! Internal LRL
; 2698        2743  3        [STAT_K_LRL_OUT]:        .CTX[COM_LRL_OUT];       ! LRL for Output
; 2699        2744  3        [STAT_K_NODES]:          .CTX[COM_STAT_NODES];    ! Nodes in sort tree
; 2700        2745  3        [STAT_K_INI_RUNS]:                                ! Initial dispersion runs
; 2701        2746  4            (IF .CTX[COM_STAT_RUNS] NEQ 0
; 2702        2747  3            THEN .CTX[COM_STAT_RUNS] ELSE .CTX[COM_RUNS]);
; 2703        2748  3        [STAT_K_MRG_ORDER]:      .CTX[COM_STAT_MERGE];    ! Maximum merge order
; 2704        2749  3        [STAT_K_MRG_PASSES]:     .CTX[COM_STAT_PASSES];   ! Number of merge passes
; 2705        2750  3        [STAT_K_WRK_ALQ]:        SOR$$WRK_ALQ();          ! Work file allocation
; 2706        2751  3        [STAT_K_MBC_INP]:                                                ! MBC for Input
; 2707        2752  3            (IF .INP EQL 0 THEN 0 ELSE .INP[DDB_RAB+RAB$B_MBC]);
; 2708        2753  3        [STAT_K_MBC_OUT]:                                                ! MBC for Output
; 2709        2754  3            (IF .OUT EQL 0 THEN 0 ELSE .OUT[DDB_RAB+RAB$B_MBC]);
; 2710        2755  3        [STAT_K_MBF_INP]:                                                ! MBF for Input
; 2711        2756  3            (IF .INP EQL 0 THEN 0 ELSE .INP[DDB_RAB+RAB$B_MBF]);
; 2712        2757  3        [STAT_K_MBF_OUT]:                                                ! MBF for Output
; 2713        2758  3            (IF .OUT EQL 0 THEN 0 ELSE .OUT[DDB_RAB+RAB$B_MBF]);
; 2714        2759  3        [INRANGE,OUTRANGE]:
; 2715        2760  3            RETURN SOR$_NYI;
; 2716        2761  2        TES);
; 2717        2762  2
; 2718        2763  2    RETURN SS$_NORMAL;
; 2719        2764  1    END;


                                        012FA          .BLKB   2
        39  35  30  2D  33  30  56  07  012FC P.AAB:   .ASCII  <7>\V03-059\                                  ;


                                    0808 00000         .ENTRY  SOR$STAT, Save R3,R11                          ; 2683
                                  7E  D4 00002         CLRL    CTX__                                          ; 2720
                    6D      0106  CF  DE 00004         MOVAL   29$, (FP)                                      ;
                    03            6C  91 00009         CMPB    (AP), #3                                       ; 2726
                                  06  1F 0000C         BLSSU   1$                                             ;
                    53        0C  AC  D0 0000E         MOVL    CONTEXT, P__                                   ;
                                  07  12 00012         BNEQ    2$                                             ;
                    53 00000000'  EF  9E 00014 1$:     MOVAB   SOR_CONTEXT, P__                               ;
        5B          63            01  CB 0001B 2$:     BICL3   #1, (P__), CTX                                 ;
                                  08  12 0001F         BNEQ    3$                                             ;
                    50 001C802C   8F  D0 00021         MOVL    #1867820, R0                                   ;
                                      04 00028         RET                                                    ;
            0130    CB            01  D0 00029 3$:     MOVL    #1, 304(CTX)                                   ;
                    02            6C  91 0002E         CMPB    (AP), #2                                       ; 2728
                                  0E  1F 00031         BLSSU   4$                                             ;
                              08  AC  D5 00033         TSTL    8(AP)                                          ;
                                  09  13 00036         BEQL    4$                                             ;
                                  6C  95 00038         TSTB    (AP)                                           ;
                                  05  13 0003A         BEQL    4$                                             ;
                              04  AC  D5 0003C         TSTL    4(AP)                                          ;
                                  08  12 0003F         BNEQ    5$                                             ;
                    50 001C80E4   8F  D0 00041 4$:     MOVL    #1868004, R0                                   ;
                                      04 00048         RET                                                    ;
                    53      009C  CB  9E 00049 5$:     MOVAB   156(R11), R3                                   ; 2732
```

```
                                51      0098   CB  9E 0004E          MOVAB   152(R11), R1                          ; 2733
                        10      00        04   BC  CF 00053          CASEL   @CODE, #0, #16                        ; 2734
        0040          0037    0031          002A     00058 6$:      .WORD   8$-6$,-                               ;
        005C          0055    004E          0047     00060                  9$-6$,-                               ;
        007E          0077    0070          0063     00068                  10$-6$,-                              ;
        00A1          009A    008F          008A     00070                  11$-6$,-                              ;
                                            0022     00078                  12$-6$,-                              ;
                                                                            13$-6$,-                              ;
                                                                            14$-6$,-                              ;
                                                                            15$-6$,-                              ;
                                                                            16$-6$,-                              ;
                                                                            17$-6$,-                              ;
                                                                            18$-6$,-                              ;
                                                                            19$-6$,-                              ;
                                                                            21$-6$,-                              ;
                                                                            22$-6$,-                              ;
                                                                            24$-6$,-                              ;
                                                                            25$-6$,-                              ;
                                                                            7$-6$                                 ;
                                50 001C8122   8F  D0 0007A 7$:      MOVL    #1868066, R0                          ; 2760
                                                  04 00081          RET                                           ;
                                50      FF72   CF  9E 00082 8$:      MOVAB   P.AAB, R0                             ; 2736
                                               54  11 00087          BRB     20$                                   ;
                                5B        7C   AB  D0 00089 9$:      MOVL    124(CTX), R11                         ; 2737
                                               77  11 0008D          BRB     28$                                   ;
                        50      7C   AB      00BC   CB  C3 0008F 10$:    SUBL3   188(CTX), 124(CTX), R0                ; 2739
                                               45  11 00096          BRB     20$                                   ;
                                5B      00C0   CB  D0 00098 11$:     MOVL    192(CTX), R11                         ; 2740
                                               67  11 0009D          BRB     28$                                   ;
                                5B      0084   CB  3C 0009F 12$:     MOVZWL  132(CTX), R11                         ; 2741
                                               60  11 000A4          BRB     28$                                   ;
                                5B      0088   CB  3C 000A6 13$:     MOVZWL  136(CTX), R11                         ; 2742
                                               59  11 000AB          BRB     28$                                   ;
                                5B      008A   CB  3C 000AD 14$:     MOVZWL  138(CTX), R11                         ; 2743
                                               52  11 000B2          BRB     28$                                   ;
                                5B      00B0   CB  D0 000B4 15$:     MOVL    176(CTX), R11                         ; 2744
                                               4B  11 000B9          BRB     28$                                   ;
                                50      00B4   CB  3C 000BB 16$:     MOVZWL  180(CTX), R0                          ; 2746
                                               1B  12 000C0          BNEQ    20$                                   ;
                                5B        7A   AB  3C 000C2          MOVZWL  122(CTX), R11                         ; 2747
                                               3E  11 000C6          BRB     28$                                   ; 2746
                                5B      00B8   CB  9A 000C8 17$:     MOVZBL  184(CTX), R11                         ; 2748
                                               37  11 000CD          BRB     28$                                   ;
                                5B      00B6   CB  3C 000CF 18$:     MOVZWL  182(CTX), R11                         ; 2749
                                               30  11 000D4          BRB     28$                                   ;
                00000000G       00             00  FB 000D6 19$:     CALLS   #0, SOR$$WRK_ALQ                      ; 2750
                                5B             50  D0 000DD 20$:     MOVL    R0, R11                               ;
                                               24  11 000E0          BRB     28$                                   ;
                                5B             63  D0 000E2 21$:     MOVL    (R3), R11                             ; 2752
                                               03  11 000E5          BRB     23$                                   ;
                                5B             61  D0 000E7 22$:     MOVL    (R1), R11                             ; 2754
                                               12  13 000EA 23$:     BEQL    26$                                   ;
                                5B        4B   AB  9A 000EC          MOVZBL  75(R11), R11                          ;
                                               14  11 000F0          BRB     28$                                   ;
                                5B             63  D0 000F2 24$:     MOVL    (R3), R11                             ; 2756
                                               07  13 000F5          BEQL    26$                                   ;
                                               09  11 000F7          BRB     27$                                   ;
```

```
                        5B          61  D0 000F9 25$:   MOVL    (R1), R11                          ; 2758
                                    04  12 000FC        BNEQ    27$                                ;
                                    5B  D4 000FE 26$:   CLRL    R11                                ;
                                    04  11 00100        BRB     28$                                ;
                        5B      4A  AB  98 00102 27$:   CVTBL   74(R11), R11                       ;
                  08    BC          5B  D0 00106 28$:   MOVL    R11, @RESULT                       ; 2734
                        50          01  D0 0010A        MOVL    #1, R0                             ; 2763
                                        04 0010D        RET                                        ; 2764
                                      0000 0010E 29$:   .WORD   Save nothing                       ; 2720
                        50      08  AC  D0 00110        MOVL    8(AP), R0                          ;
                        50      04  A0  D0 00114        MOVL    4(R0), R0                          ;
                                FC  A0  9F 00118        PUSHAB  CTX__                              ;
                                    01  DD 0011B        PUSHL   #1                                 ;
                                    5E  DD 0011D        PUSHL   SP                                 ;
                        7E      04  AC  7D 0011F        MOVQ    4(AP), -(SP)                       ;
                EBF2    CF          03  FB 00123        CALLS   #3, COND_HAND                      ;
                                        04 00128        RET                                        ;

; Routine Size:  297 bytes,    Routine Base:  SOR$RO_CODE + 1304


; 2720          2765  1
; 2721          2766  1 END
; 2722          2767  0 ELUDOM




;                                PSECT SUMMARY
;
;       Name                    Bytes                       Attributes
;
;   SOR$RW_PICDATA                  4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   SOR$RO_CODE                  5165  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   .  ABS  .                       0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)
;   SOR$RO_CODE___________________1 0  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   SOR$RO_CODE___________________3 0  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



;                          Library Statistics
;
;                                  -------- Symbols --------      Pages       Processing
;       File                       Total    Loaded   Percent      Mapped      Time
;
;   _$255$DUA28:[SYSLIB]STARLET.L32;1   9776       97         0       581         00:00.9
;   _$255$DUA28:[SORT32.SRC]SORLIB.L32;1  409      192        46        34         00:00.4
```

```
;                                       COMMAND QUALIFIERS

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

; Size:         5147 code + 22 data bytes
; Run Time:       01:42.7
; Elapsed Time:   05:24.1
; Lines/CPU Min:   1616
; Lexemes/CPU-Min: 28359
; Memory Used:  453 pages
; Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SORINTERF
LIS

SORKEYSUB
LIS

SORFILNAM
LIS

SORCOMPAR
LIS

SORENTRY
LIS