```
   SSSSSSSSSSS     000000000     RRRRRRRRRRR     TTTTTTTTTTTTTTT     333333333       222222222
   SSSSSSSSSSS     000000000     RRRRRRRRRRRR    TTTTTTTTTTTTTTT     333333333       222222222
   SSSSSSSSSSS     000000000     RRRRRRRRRRRR    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          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  RRRRRRRRRRRR           TTT                333                   222
   SSSSSSSSS     000          000  RRRRRRRRRRRR           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
            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
```

FILEID**SORCOMMAN

```
  SSSSSSSS    000000    RRRRRRRR      CCCCCCCC    000000    MM      MM  MM      MM    AAAAAA    NN      NN
  SSSSSSSS    000000    RRRRRRRR      CCCCCCCC    000000    MM      MM  MM      MM    AAAAAA    NN      NN
SS          00      00  RR      RR  CC          00      00  MMMM  MMMM  MMMM  MMMM  AA      AA  NN      NN
SS          00      00  RR      RR  CC          00      00  MMMM  MMMM  MMMM  MMMM  AA      AA  NN      NN
SS          00      00  RR      RR  CC          00      00  MM  MM  MM  MM  MM  MM  AA      AA  NNNN    NN
SS          00      00  RR      RR  CC          00      00  MM  MM  MM  MM  MM  MM  AA      AA  NNNN    NN
  SSSSSS    00      00  RRRRRRRR    CC          00      00  MM      MM  MM      MM  AA      AA  NN  NN  NN
  SSSSSS    00      00  RRRRRRRR    CC          00      00  MM      MM  MM      MM  AA      AA  NN  NN  NN
        SS  00      00  RR  RR      CC          00      00  MM      MM  MM      MM  AAAAAAAAAA  NN    NNNN
        SS  00      00  RR  RR      CC          00      00  MM      MM  MM      MM  AAAAAAAAAA  NN    NNNN
        SS  00      00  RR    RR    CC          00      00  MM      MM  MM      MM  AA      AA  NN      NN      ....
        SS  00      00  RR    RR    CC          00      00  MM      MM  MM      MM  AA      AA  NN      NN      ....
SSSSSSSS      000000    RR      RR    CCCCCCCC    000000    MM      MM  MM      MM  AA      AA  NN      NN      ....
SSSSSSSS      000000    RR      RR    CCCCCCCC    000000    MM      MM  MM      MM  AA      AA  NN      NN      ....

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$COMMAND (
;     2          0002  0                 IDENT = 'V04-000'       ! File: SORCOMMAN.B32 Edit: PDG3024
;     3          0003  0                 ) =
;     4          0004  1 BEGIN
;     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 processes the SORT/MERGE command line.
;    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 !       T03-016 Added a check that the CLI$K_SORT_xxx and CLI$K_MERG_xxx
;    46          0046  1 !               externals have the same value.  PDG 9-Dec-1982
;    47          0047  1 !       T03-017 Remove reference to CLI$END_PARSE.  PDG 22-Dec-1982
;    48          0048  1 !       T03-018 Stable option is allowed for merges.  PDG 27-Jan-1983
;    49          0049  1 !       T03-019 Change DEC_MULTINATIONAL to MULTINATIONAL.  PDG 13-Apr-1983
;    50          0050  1 !       T03-020 Various changes for KANJI.  PDG 2-May-1983
;    51          0051  1 !       T03-021 Make SOR$$DTYPE_T_W a weak external.  PDG 19-May-1983
;    52          0052  1 !       T03-022 Convert to use new CLI interface.  PDG 10-Aug-1983
;    53          0053  1 !       T03-023 New CLI interface for checkpointing.  PDG 15-Dec-1983
;    54          0054  1 !       T03-024 Conditionalize checkpointing code.  PDG 07-Jan-1984
;    55          0055  1 !--
```

```
;    57          0056  1 LIBRARY 'SYS$LIBRARY:LIB';
;    58          0057  1 REQUIRE 'SRC$:SORMSG';
;    59          0234  1 !+
;    60          0235  1 !
;    61          0236  1 ! Note that we do not use SORLIB.REQ.  This ensures that the code in this
;    62          0237  1 ! module does not depend on internal data structures.
;    63          0238  1 !
;    64          0239  1 !-
;    65          0240  1
;    66          0241  1 ! These literals control support for user-defined key data types,
;    67          0242  1 ! and checkpointing.
;    68          0243  1 !
;    69          0244  1 LITERAL
;    70          0245  1     FUN_K_KANJI = %VARIANT,
;    71          0246  1     FUN_K_CHECKPOINT = %VARIANT;
;    72          0247  1
;    73          0248  1
;    74          0249  1 FORWARD ROUTINE
;    75          0250  1     CMD_ERROR,                                  ! Issue an error diagnostic
;    76          0251  1     SOR$$COMMAND,                               ! Parse the command line
;    77          0252  1     NUMERIC_PARSE,                              ! Parse a number
;    78          0253  1     SET_SPEC:           NOVALUE,                ! Parse specification file name
;    79          0254  1     SET_COLL:           NOVALUE,                ! Get argument of /COLLATING_SEQUE=
;    80          0255  1     SET_BUCK:           NOVALUE,                ! Get the number of buckets
;    81          0256  1     SET_ALLO:           NOVALUE,                ! Get output file allocation
;    82          0257  1     SET_OUT_FMT:        NOVALUE,                ! Set output file record format
;    83          0258  1     SET_INP_FMT:        NOVALUE,                ! Set input record format
;    84          0259  1     SET_KEY:            NOVALUE,                ! Parse and store key definitions
;    85          0260  1     LOOKUP_KEY,                                 ! Compare keyword strings
;    86          0261  1     !
;    87          0262  1     ! Define the SORT-specific routines
;    88          0263  1     !
;    89          0264  1     SET_PROC:           NOVALUE,                ! Set type of process requested
;    90          0265  1     NOT_WORK:           NOVALUE,                ! Set no work files
;    91        U 0266  1 %IF FUN_K_KANJI %THEN
;    92        U 0267  1     DEL_BOTH,
;    93        U 0268  1     SET_DUPL:           NOVALUE,
;    94          0269  1 %FI
;    95          0270  1     SET_WORK:           NOVALUE;                ! Get the number of work files
;    96          0271  1
;    97          0272  1
;    98        U 0273  1 %IF FUN_K_CHECKPOINT %THEN
;    99        U 0274  1 EXTERNAL ROUTINE
;   100        U 0275  1     !
;   101        U 0276  1     ! Routines that interface to the CLI
;   102        U 0277  1     !
;   103        U 0278  1     CLI_BEGIN:          NOVALUE,                ! Setup CLI processing
;   104        U 0279  1     CLI_GET_VALUE,                              ! Get value of item
;   105        U 0280  1     CLI_PRESENT,                                ! Determine if item is present
;   106        U 0281  1     CLI_NEXT_QUAL,                              ! Get next qualifier
;   107        U 0282  1     CLI_END:            NOVALUE;                ! End CLI processing
;   108          0283  1 %ELSE
;   109          0284  1 EXTERNAL ROUTINE
;   110          0285  1     !
;   111          0286  1     ! Routines that interface to the CLI
;   112          0287  1     !
;   113          0288  1     CLI$GET_VALUE,                              ! Get value of item
```

```
;  114          0289  1      CLI$PRESENT,                        ! Determine if item is present
;  115          0290  1      CLI$NEXT_QUAL;                      ! Get next qualifier
;   16          0291  1  BIND ROUTINE
;  117          0292  1      CLI_GET_VALUE       = CLI$GET_VALUE,
;  118          0293  1      CLI_PRESENT         = CLI$PRESENT,
;  119          0294  1      CLI_NEXT_QUAL       = CLI$NEXT_QUAL;
;  120          0295  1  %FI
;  121          0296  1
;  122          0297  1  MACRO
;  123          0298  1      ! Macro to test an assertion about compile-time constants.
;  124          0299  1      !
;  125        M 0300  1      ASSERT_(A)=
;  126        M 0301  1          %IF NOT (A)
;  127        M 0302  1          %THEN
;  128        M 0303  1              %ERROR('Assertion failed')
;  129          0304  1          %FI %;
```

```
;  131          0305  1 ! These macroes define the external and internal representations of options for
;  132          0306  1 ! command line qualifiers.  The second parameter in each pair may be translated;
;  133          0307  1 ! the first, however, is used to define the internal name for this option, and
;  134          0308  1 ! may not be translated.
;  135          0309  1 !
;  136          0310  1 MACRO
;  137        M 0311  1     STR_OPT_OUTFMT =                        ! outfile/FORMAT=(...)
;  138        M 0312  1          'FIXE', 'FIXED',
;  139        M 0313  1          'VARI', 'VARIABLE',
;  140        M 0314  1          'CONT', 'CONTROLLED',
;  141        M 0315  1          'SIZE', 'SIZE',
;  142          0316  1          'BLOC', 'BLOCK_SIZE' %,
;  143          0317  1
;  144        M 0318  1     STR_OPT_INPFMT =                        ! inpfile/FORMAT=(...)
;  145        M 0319  1          'FILE', 'FILE_SIZE',
;  146          0320  1          'RECO', 'RECORD_SIZE' %,
;  147          0321  1
;  148        M 0322  1     STR_OPT_PROCESS =                       ! /PROCESS=...
;  149        M 0323  1          'RECO', 'RECORD',
;  150        M 0324  1          'TAG', 'TAG',
;  151        M 0325  1          'ADDR', 'ADDRESS',
;  152          0326  1          'INDE', 'INDEX' %,
;  153          0327  1
;  154        M 0328  1     STR_OPT_KEY =                           ! /KEY=...
;  155        M 0329  1          'ASCE', 'ASCENDING',
;  156        M 0330  1          'BINA', 'BINARY',
;  157        M 0331  1          'CHAR', 'CHARACTER',
;  158        M 0332  1          'DECI', 'DECIMAL',
;  159        M 0333  1          'DESC', 'DESCENDING',
;  160        M 0334  1 %IF FUN_K_KANJI %THEN
;  161        M 0335  1          'DTYP', 'DTYPE',
;  162        M 0336  1 %FI
;  163        M 0337  1          'F_FL', 'F_FLOATING',
;  164        M 0338  1          'D_FL', 'D_FLOATING',
;  165        M 0339  1          'G_FL', 'G_FLOATING',
;  166        M 0340  1          'H_FL', 'H_FLOATING',
;  167        M 0341  1          'LEAD', 'LEADING_SIGN',
;  168        M 0342  1          'NUMB', 'NUMBER',                  ! NUMBER:nn
;  169        M 0343  1          'OVER', 'OVERPUNCHED_SIGN',
;  170        M 0344  1          'POSI', 'POSITION',                ! POSITION:nn
;  171        M 0345  1          'PACK', 'PACKED_DECIMAL',
;  172        M 0346  1          'SI', 'SI',                        ! SI:nn or SIGNED
;  173        M 0347  1          'SIGN', 'SIGNED',
;  174        M 0348  1          'SIZE', 'SIZE',                    ! SIZE:nn
;  175        M 0349  1          'SEPA', 'SEPARATE_SIGN',
;  176        M 0350  1          'TRAI', 'TRAILING_SIGN',
;  177        M 0351  1          'UNSI', 'UNSIGNED',
;  178          0352  1          'ZONE', 'ZONED' %,
;  179          0353  1
;  180        M 0354  1     STR_OPT_COLL =
;  181        M 0355  1          'ASCI', 'ASCII',
;  182        M 0356  1          'EBCD', 'EBCDIC',
;  183          0357  1          'MULT', 'MULTINATIONAL' %;
;  184          0358  1
;  185          0359  1
;  186          0360  1 ! Default specification file
;  187          0361  1 !
```

```
;  188          0362  1 MACRO
;  189          0363  1     STR_DEF_SPECFILE =  'SYS$INPUT' %;
```

```
;  191          0364  1 EXTERNAL LITERAL
;  192          0365  1     SOR$M_STABLE,
;  193          0366  1     SOR$M_SEQ_CHECK,
;  194          0367  1     SOR$M_NODUPS,
;  195          0368  1     SOR$M_EBCDIC,
;  196          0369  1     SOR$M_MULTI,
;  197          0370  1     SOR$M_LOAD_FILL;
;  198          0371  1
;  199          0372  1 EXTERNAL LITERAL
;  200          0373  1     SOR$GK_RECORD,
;  201          0374  1     SOR$GK_TAG,
;  202          0375  1     SOR$GK_ADDRESS,
;  203          0376  1     SOR$GK_INDEX;
;  204          0377  1
;  205          0378  1 ! Define the maximum number of keys we will allow.  This need not agree with
;  206          0379  1 ! the value used by the callable interface routines, but it should be as large.
;  207          0380  1 !
;  208          0381  1 LITERAL
;  209          0382  1     MAX_KEYS = 255;
;  210          0383  1
;  211          0384  1 MACRO
;  212          0385  1     ELIF =      ELSE IF %,
;  213          0386  1     BASE_ =     0, 0, 0, 0 %,
;  214          0387  1     W0_=        0, 16,0 %,
;  215          0388  1     W1_=        16,16,0 %,
;  216          0389  1     SWT_=       16,16,1 %;
;  217          0390  1
;  218          0391  1 MACRO
;  219          0392  1     ! Fields for key buffer.
;  220          0393  1     !
;  221          0394  1     KEY_NUMBER=         0, W0_ %,       ! Number of key descriptions
;  222          0395  1     KEY_KBF(N)=         2*(N),16,0,0 %, ! Address of KBF (0..number-1)
;  223          0396  1     KBF_TYPE=           0, W0_ %,       ! Data type of key
;  224          0397  1     KBF_ORDER=          0, W1_ %,       ! True iff descending order
;  225          0398  1     KBF_ORDER_S=        0, SWT_ %,      ! Signed version of KBF_ORDER
;  226          0399  1     KBF_POSITION=       1, W0_ %,       ! Offset to key within record (1..LRL)
;  227          0400  1     KBF_LENGTH=         1, W1_ %;       ! Length of key
;  228          0401  1 LITERAL
;  229          0402  1     KBF_S_ENTRY=        2;              ! Size in longwords of key description
;  230          0403  1
;  231          0404  1
;  232        L 0405  1 %IF NOT %DECLARED(SORT$_FACILITY)
;  233        U 0406  1 %THEN
;  234        U 0407  1     LITERAL
;  235        U 0408  1         SORT$_FACILITY = SOR$_FACILITY;
;  236        U 0409  1     UNDECLARE
;  237        U 0410  1         SOR$_FACILITY;
;  238          0411  1     %FI
;  239          0412  1 MACRO
;  240        M 0413  1     DEFSHR [MSG,SEV] =
;  241        M 0414  1         %NAME('SOR$_SHR_',MSG) =
;  242        M 0415  1             %NAME('SHR$_',MSG) +
;  243          0416  1             %NAME('STS$K_',SEV) + SORT$_FACILITY ^ 16 %;
;  244          0417  1 LITERAL
;  245        P 0418  1     DEFSHR (
;  246        P 0419  1         BADBYTE,  SEVERE,       ! Invalid data (!XB) at !XL
;  247        P 0420  1         BADKEY,   SEVERE,       ! !AS is an invalid keyword
```

```
;  248        P 0421  1          BADLOGIC, SEVERE,       ! Internal logic error detected
;  249        P 0422  1          BADWORD,  SEVERE,       ! Invalid data (!XW) at !XL
;  250        P 0423  1          NOVALUE,  SEVERE,       ! '!AS' keyword requires a value
;  251        P 0424  1          SYNTAX,   SEVERE,       ! Error parsing !AS
;  252          0425  1          SYSERROR, SEVERE);      ! System service error
;  253          0426  1 LITERAL
;  254          0427  1     SOR$_SHR_BADKEY_W =
;  255          0428  1          SHR$_BADKEY + STS$K_WARNING + SORT$_FACILITY ^ 16;
```

```
;  257          0429  1 ! This macro builds a keyword description table, passed to LOOKUP_KEY
;  258          0430  1 !
;  259          0431  1 MACRO
;  260        M 0432  1     KEY_TABLE_[] =
;  261        M 0433  1         LITERAL TABLE_SIZE = %LENGTH/2,
;  262        M 0434  1                 KEY_TABLE1_(%REMAINING);
;  263          0435  1         BIND    KEYWORD_TABLE = PLIT(KEY_TABLE2_(%REMAINING)): VECTOR %,
;  264          0436  1     KEY_TABLE1_[X,Y] = %NAME('K_',X) = %COUNT %,
;  265          0437  1     KEY_TABLE2_[X,Y] = UPLIT BYTE(%ASCIC Y) %;
;  266          0438  1
;  267          0439  1
;  268          0440  1 ! This macro stores a value in a variable.  It checks for overflow.
;  269          0441  1 !
;  270          0442  1 MACRO
;  271        M 0443  1     STORE_(X,Y) =
;  272        M 0444  1         %IF %ALLOCATION(X) LSS %UPVAL
;  273        M 0445  1         %THEN
;  274        M 0446  1             BEGIN
;  275        M 0447  1             LOCAL
;  276        M 0448  1                 Z;      ! So Y is evaluated only once; a BIND may be better
;  277        M 0449  1             Z = (Y);
;  278        M 0450  1             X = .Z;
;  279        M 0451  1             IF .Z<0,%ALLOCATION(X)*%BPUNIT,0> NEQ .Z
;  280        M 0452  1             THEN
;  281        M 0453  1                 RETURN CMD_ERROR(
;  282        M 0454  1                     %IF %ALLOCATION(X) EQL 1
;  283        M 0455  1                     %THEN SOR$_SHR_BADBYTE
;  284        M 0456  1                     %ELSE SOR$_SHR_BADWORD
;  285        M 0457  1                     %FI, 2, .Z, 0);
;  286        M 0458  1             END
;  287        M 0459  1         %ELSE
;  288        M 0460  1             X = (Y)
;  289          0461  1         %FI %;
;  290          0462  1
;  291          0463  1
;  292          0464  1 ! Define these mnemonic literals
;  293          0465  1 !
;  294          0466  1 LITERAL
;  295          0467  1     TRUE =              1,
;  296          0468  1     FALSE =             0;
;  297          0469  1
;  298          0470  1
;  299          0471  1 ! Define the ONEOF macro, and its related macroes.
;  300          0472  1 !
;  301          0473  1 MACRO
;  302          0474  1     XBMSK_[A]= (1 ^ (31 - (A))) %,
;  303          0475  1     BMSK_[]= (0 OR XBMSK_(%REMAINING)) %,
;  304          0476  1     ONEOF_(A,B)= (((B) ^ (A)) LSS 0) %;
```

```
;  306          0477  1 ! Define bit positions within FLAGS
;  307          0478  1 !
;  308          0479  1 LITERAL
;  309          0480  1     IOS_SORT=           1,      ! True if we were invoked for SORT
;  310          0481  1     IOS_SEQU=           2,      ! /SEQUENTIAL
;  311          0482  1     IOS_RELA=           3,      ! /RELATIVE
;  312          0483  1     IOS_INDE=           4,      ! /INDEXED
;  313          0484  1     IOS_OVER=           5,      ! /OVERLAY
;  314          0485  1     IOS_BUCK=           6,      ! /BUCKET
;  315          0486  1     IOS_ALLO=           7,      ! /ALLOCATION
;  316          0487  1     IOS_WORK=           8,      ! /WORK_FILES
;  317          0488  1     IOS_CONT=           9,      ! /CONTIGUOUS
;  318          0489  1     IOS_STAB=           10,     ! /STABLE
;  319          0490  1     IOS_STAT=           11,     ! /STATISTICS
;  320          0491  1     IOS_LOAD=           12,     ! /LOAD
;  321          0492  1     IOS_CHEC=           13,     ! /CHECK
;  322          0493  1     IOS_DUPL=           14,     ! /DUPLICATES
;  323          0494  1     IOS_KEY=            15,     ! /KEY
;  324          0495  1     IOS_PROC=           16,     ! /PROCESS
;  325          0496  1     IOS_BOTH=           17,     ! /DUPLICATES=DELBOTH
;  326          0497  1     IOS_CHKP=           18;     ! /CHKPNT
```

```
;  328          0498  1 ! Define the codes for the sort or merge qualifiers
;  329          0499  1 ! Note that they both use the the n  's CLI$K_SORT_xxx.
;  330          0500  1 ! If any SORT-specific qualifiers are used with MERGE, or any MERGE-specific
;  331          0501  1 ! qualifiers are used with SORT, the code will signal CLI$_INVQUAL.
;  332          0502  1 !
;  333          0503  1 MACRO
;  334        M 0504  1     QUALCODES =
;  335        M 0505  1               !
;  336        M 0506  1               ! The SORT-specific qualifiers
;  337        M 0507  1               !
;  338        M 0508  1               'PROCESS',              ! /PROCESS=sort-type
;  339        M 0509  1               'WORK_FILES',           ! /WORK_FILES=n
;  340        M 0510  1               !
;  341        M 0511  1               ! The MERGE-specific qualifiers
;  342        M 0512  1               !
;  343        M 0513  1               'CHECK_SEQUENCE',       ! /CHECK_SEQUENCE
;  344        M 0514  1               !
;  345        M 0515  1               ! The qualifiers common to both SORT and MERGE
;  346        M 0516  1               !
;  347        M 0517  1               'KEY',                  ! /KEY=sort-key
;  348        M 0518  1               'SPECIFICATION',        ! /SPECIFICATION=spec-file
;  349        M 0519  1               'STABLE',               ! /STABLE
;  350        M 0520  1               'COLLATING_SEQUENCE',   ! /COLLATING_SEQUENCE=collating-type
;  351        M 0521  1               'STATISTICS',           ! /STATISTICS
;  352        M 0522  1               'DUPLICATES',           ! /DUPLICATES
;  353        M 0523  1 %IF FUN_K_CHECKPOINT %THEN
;  354        M 0524  1               'CHKPNT',               ! /CHKPNT
;  355        M 0525  1 %FI
;  356        M 0526  1               !
;  357        M 0527  1               ! Output file qualifiers
;  358        M 0528  1               !
;  359        M 0529  1               'SEQUENTIAL',           ! /SEQUENTIAL
;  360        M 0530  1               'RELATIVE',             ! /RELATIVE
;  361        M 0531  1               'INDEXED_SEQUENTIAL',   ! /INDEXED
;  362        M 0532  1 !!!!!!! 'FORMAT',                     ! /FORMAT=record-format (handled specially)
;  363        M 0533  1               'BUCKET_SIZE',          ! /BUCKET_SIZE=n
;  364        M 0534  1               'ALLOCATION',           ! /ALLOCATION=n
;  365        M 0535  1               'CONTIGUOUS',           ! /CONTIGUOUS
;  366        M 0536  1               'OVERLAY',              ! /OVERLAY
;  367          0537  1               'LOAD_FILL' %;          ! /LOAD_FILL
;  368          0538  1 MACRO
;  369        M 0539  1     QC_4[X] =
;  370        M 0540  1               %IF %CHARCOUNT(X) LEQ 4 %THEN X %ELSE %EXACTSTRING(4,' ',X) %FI,
;  371          0541  1               %STRING(X) %,
;  372          0542  1     QC_1[X,Y] = %NAME('SD_',X) = $DESCRIPTOR(Y): $BBLOCK[DSC$C_S_BLN] %,
;  373        M 0543  1     QC_2[X,Y] =
;  374        M 0544  1               %NAME('SD_',X),
;  375        M 0545  1               %IF %DECLARED(%NAME('IOS_',X)) %THEN %NAME('IOS_',X) %ELSE 0 %FI,
;  376        M 0546  1               %IF %DECLARED(%NAME('SET_',X)) %THEN %NAME('SET_',X) %ELSE 0 %FI,
;  377          0547  1               %IF %DECLARED(%NAME('NOT_',X)) %THEN %NAME('NOT_',X) %ELSE 0 %FI %,
;  378          0548  1     QC_3[] = %LENGTH %;
;  379          0549  1 LITERAL
;  380          0550  1     STR_K_QUALCODES = QC_3( QUALCODES );
;  381          0551  1 MACRO
;  382          0552  1     STR_OPT_QUALCODES = %UNQUOTE QC_4( QUALCODES ) %;
;  383          0553  1 BIND
;  384          0554  1     QC_1( STR_OPT_QUALCODES ),
```

```
;  385        P 0555  1      QC_1( QC_4(
;  386        P 0556  1          'P1',           ! Input file(s)
;  387        P 0557  1          'P2',           ! Output file
;  388        P 0558  1          '$VERB',        ! Verb from the command line
;  389          0559  1          'FORMAT' ) ),   ! /FORMAT=record-format
;  390          0560  1      CLIQUALDESC = UPLIT( QC_2( STR_OPT_QUALCODES ) ): VECTOR;
;  391          0561  1
;  392          0562  1  OWN
;  393          0563  1      ! Qualifier string value descriptor block
;  394          0564  1      !
;  395          0565  1      STRING_DESC:        $BBLOCK[DSC$C_D_BLN],
;  396          0566  1      SUBVAL_DESC:        $BBLOCK[DSC$C_S_BLN];
;  397          0567  1
;  398          0568  1
;  399          0569  1  EXTERNAL ROUTINE
;  400          0570  1      CLI$PRESENT:        ADDRESSING_MODE(GENERAL),       ! Determine if /CHKPNT
;  401          0571  1      STR$COPY_R:         ADDRESSING_MODE(GENERAL),       ! String copy
;  402          0572  1      SOR$SPEC_FILE:      ADDRESSING_MODE(GENERAL),       ! Process spec file
;  403          0573  1      SOR$PASS_FILES:     ADDRESSING_MODE(GENERAL),       ! Pass file names
;  404          0574  1      SOR$BEGIN_SORT:     ADDRESSING_MODE(GENERAL),       ! Initialize the sort
;  405          0575  1      SOR$BEGIN_MERGE:    ADDRESSING_MODE(GENERAL),       ! Initialize the merge
;  406        U 0576  1  %IF FUN_K_KANJI %THEN
;  407        U 0577  1      SOR$$DTYPE_T_W:     ADDRESSING_MODE(GENERAL) WEAK,  ! Key name to code
;  408          0578  1  %FI
;  409          0579  1      LIB$SIGNAL:         ADDRESSING_MODE(GENERAL);       ! Signal an error
;  410          0580  1
;  411          0581  1  EXTERNAL LITERAL
;  412          0582  1      CLI$_VALCNVERR,     ! Error converting value
;  413          0583  1      CLI$_IVCHAR,        ! Invalid character
;  414          0584  1      CLI$_IVVALU,        ! Invalid value
;  415          0585  1      CLI$_NEGATED,       ! Qualifier was explicitly negated
;  416          0586  1      CLI$_INVQUAL;       ! SORT-specific quals used with MERGE, or vica-versa.
```

```
;  418          0587  1 OWN
;  419          0588  1     CMD_CLEAR0: VECTOR[0],
;  420          0589  1     !
;  421          0590  1     ! Parameters passed to PASS_FILES
;  422          0591  1     !
;  423          0592  1     ORG:        BYTE,           ! File organization
;  424          0593  1     RFM:        BYTE,           ! Record format
;  425          0594  1     BKS:        BYTE,           ! Bucket size
;  426          0595  1     BLS:        WORD,           ! Block size
;  427          0596  1     MRS:        WORD,           ! Maximum record size
;  428          0597  1     OUT_ALQ:    LONG,           ! Allocation quantity
;  429          0598  1     FOP:        LONG,           ! File-processing options
;  430          0599  1     !
;  431          0600  1     ! Parameters returned from SOR$$COMMAND
;  432          0601  1     !
;  433          0602  1     KBF:        REF BLOCK,      ! Key buffer address
;  434          0603  1     LRL:        WORD,           ! Longest record length
;  435          0604  1     INP_ALQ:    LONG,           ! Input file allocation
;  436          0605  1     WRK:        BYTE,           ! Number of work files
;  437          0606  1     TYP:        BYTE,           ! Sort type
;  438          0607  1     OPT:        LONG,           ! Options
;  439          0608  1     CTX:        LONG,           ! Address of context longword
;  440          0609  1     FSZ:        BYTE,           ! Size of the VFC field (word suffices)
;  441          0610  1     !
;  442          0611  1     ! Sort parser flags
;  443          0612  1     !
;  444          0613  1     FLAGS:      BITVECTOR[32],
;  445          0614  1     !
;  446          0615  1     CMD_CLEAR1: VECTOR[0];
```

```
;  448          0616  1 ROUTINE CMD_ERROR(ERR) =
;  449          0617  1
;  450          0618  1 !++
;  451          0619  1 !
;  452          0620  1 ! FUNCTIONAL DESCRIPTION:
;  453          0621  1 !
;  454          0622  1 !       This routine signals an error diagnostic.
;  455          0623  1 !
;  456          0624  1 ! FORMAL PARAMETERS:
;  457          0625  1 !
;  458          0626  1 !       Parameters passed to LIB$SIGNAL.
;  459          0627  1 !
;  460          0628  1 ! IMPLICIT INPUTS:
;  461          0629  1 !
;  462          0630  1 !       NONE
;  463          0631  1 !
;  464          0632  1 ! IMPLICIT OUTPUTS:
;  465          0633  1 !
;  466          0634  1 !       NONE
;  467          0635  1 !
;  468          0636  1 ! ROUTINE VALUE:
;  469          0637  1 !
;  470          0638  1 !       System status (first parameter of signalled status), with the
;  471          0639  1 !       INHIB_MSG bit set.
;  472          0640  1 !
;  473          0641  1 ! SIDE EFFECTS:
;  474          0642  1 !
;  475          0643  1 !       The image may be exitted due to the error.
;  476          0644  1 !
;  477          0645  1 !--
;  478          0646  2     BEGIN
;  479          0647  2     BUILTIN
;  480          0648  2         AP,
;  481          0649  2         CALLG;
;  482          0650  2     CALLG(.AP, LIB$SIGNAL);
;  483          0651  2     RETURN .ERR OR STS$M_INHIB_MSG;
;  484          0652  1     END;


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

                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                53  53  45  43  4F  52  50  00000 P.AAB:  .ASCII  \PROCESS\                    ;
                                                        00007           .BLKB   1                               ;
                                                    00000007    00008 P.AAA:  .LONG   7                         ;
                                                    00000000'   0000C         .ADDRESS P.AAB                    ;
                            53  45  4C  49  46  5F  4B  52  4F  57  00010 P.AAD:  .ASCII  \WORK_FILES\           ;
                                                        0001A           .BLKB   2                               ;
                                                    0000000A    0001C P.AAC:  .LONG   10                        ;
                                                    00000000'   00020         .ADDRESS P.AAD                    ;
    45  43  4E  45  55  51  45  53  5F  4B  43  45  48  43  00024 P.AAF:  .ASCII  \CHECK_SEQUENCE\              ;
                                                        00032           .BLKB   2                               ;
                                                    0000000E    00034 P.AAE:  .LONG   14                        ;
                                                    00000000'   00038         .ADDRESS P.AAF                    ;
                                                59  45  4B  0003C P.AAH:  .ASCII  \KEY\                       ;
```

```
                                                  0003F            .BLKB   1
                                      00000003    00040 P.AAG:     .LONG   3
                                      00000000'   00044            .ADDRESS P.AAH
                                  4E  50  53      00048 P.AAJ:     .ASCII  \SPECIFICATION\
                                                  00055            .BLKB   3
                                      0000000D    00058 P.AAI:     .LONG   13
                                      00000000'   0005C            .ADDRESS P.AAJ
                                  45  54  53      00060 P.AAL:     .ASCII  \STABLE\
                                                  00066            .BLKB   2
                                      00000006    00068 P.AAK:     .LONG   6
                                      00000000'   0006C            .ADDRESS P.AAL
45  55  51  45  53  5F  47  4E  49  54  41  4C  4C  4F  43  00070 P.AAN:     .ASCII  \COLLATING_SEQUENCE\
                                                  45  43  4E  0007F
                                                  00082            .BLKB   2
                                      00000012    00084 P.AAM:     .LONG   18
                                      00000000'   00088            .ADDRESS P.AAN
                                  53  54  41  54  53  0008C P.AAP:     .ASCII  \STATISTICS\
                                                  00096            .BLKB   2
                                      0000000A    00098 P.AAO:     .LONG   10
                                      00000000'   0009C            .ADDRESS P.AAP
                                  53  45  54  41  43  49  4C  50  55  44  000A0 P.AAR:     .ASCII  \DUPLICATES\
                                                  000AA            .BLKB   2
                                      0000000A    000AC P.AAQ:     .LONG   10
                                      00000000'   000B0            .ADDRESS P.AAR
                                  4C  41  49  54  4E  45  55  51  45  53  000B4 P.AAT:     .ASCII  \SEQUENTIAL\
                                                  000BE            .BLKB   2
                                      0000000A    000C0 P.AAS:     .LONG   10
                                      00000000'   000C4            .ADDRESS P.AAT
                                          45  56  49  54  41  4C  45  52  000C8 P.AAV:     .ASCII  \RELATIVE\
                                      00000008    000D0 P.AAU:     .LONG   8
                                      00000000'   000D4            .ADDRESS P.AAV
54  4E  45  55  51  45  53  5F  44  45  58  45  44  4E  49  000D8 P.AAX:     .ASCII  \INDEXED_SEQUENTIAL\
                                                  4C  41  49  000E7
                                                  000EA            .BLKB   2
                                      00000012    000EC P.AAW:     .LONG   18
                                      00000000'   000F0            .ADDRESS P.AAX
                              45  5A  49  53  5F  54  45  4B  43  55  42  000F4 P.AAZ:     .ASCII  \BUCKET_SIZE\
                                                  000FF            .BLKB   1
                                      0000000B    00100 P.AAY:     .LONG   11
                                      00000000'   00104            .ADDRESS P.AAZ
                                  4E  4F  49  54  41  43  4F  4C  4C  41  00108 P.ABB:     .ASCII  \ALLOCATION\
                                                  00112            .BLKB   2
                                      0000000A    00114 P.ABA:     .LONG   10
                                      00000000'   00118            .ADDRESS P.ABB
                                  53  55  4F  55  47  49  54  4E  4F  43  0011C P.ABD:     .ASCII  \CONTIGUOUS\
                                                  00126            .BLKB   2
                                      0000000A    00128 P.ABC:     .LONG   10
                                      00000000'   0012C            .ADDRESS P.ABD
                                          59  41  4C  52  45  56  4F  00130 P.ABF:     .ASCII  \OVERLAY\
                                                  00137            .BLKB   1
                                      00000007    00138 P.ABE:     .LONG   7
                                      00000000'   0013C            .ADDRESS P.ABF
                                      4C  4C  49  46  5F  44  41  4F  4C  00140 P.ABH:     .ASCII  \LOAD_FILL\
                                                  00149            .BLKB   3
                                      00000009    0014C P.ABG:     .LONG   9
                                      00000000'   00150            .ADDRESS P.ABH
                                                    31  50  00154 P.ABJ:     .ASCII  \P1\
```

```
                                 00156          .BLKB   2
                        00000002 00158 P.ABI:   .LONG   2
                        00000000' 0015C         .ADDRESS P.ABJ
                          32  50  00160 P.ABL:   .ASCII  \P2\
                                 00162          .BLKB   2
                        00000002 00164 P.ABK:   .LONG   2
                        00000000' 00168         .ADDRESS P.ABL
                    42  52  45  56  24  00160 P.ABN:   .ASCII  \$VERB\
                                 00171          .BLKB   3
                        00000005 00174 P.ABM:   .LONG   5
                        00000000' 00178         .ADDRESS P.ABN
                54  41  4D  52  4F  46  0017C P.ABP:   .ASCII  \FORMAT\
                                 00182          .BLKB   2
                        00000006 00184 P.ABO:   .LONG   6
                        00000000' 00188         .ADDRESS P.ABP
                        00000000' 0018C P.ABQ:  .ADDRESS SD_PROC
                        00000010 00190          .LONG   16
                        00000000V 00194         .ADDRESS SET_PROC
                        00000000 00198          .LONG   0
                        00000000' 0019C         .ADDRESS SD_WORK
                        00000008 001A0          .LONG   8
00000000' 00000000V 00000000V 001A4             .ADDRESS SET_WORK, NOT_WORK, SD_CHEC
00000000  00000000  0000000D  001B0             .LONG   13, 0, 0
                        00000000' 001BC         .ADDRESS SD_KEY
                        0000000F 001C0          .LONG   15
                        00000000V 001C4         .ADDRESS SET_KEY
                        00000000 001C8          .LONG   0
                        00000000' 001CC         .ADDRESS SD_SPEC
                        00000000 001D0          .LONG   0
                        00000000V 001D4         .ADDRESS SET_SPEC
                        00000000 001D8          .LONG   0
                        00000000' 001DC         .ADDRESS SD_STAB
00000000  00000000  0000000A 001E0              .LONG   10, 0, 0
                        00000000' 001EC         .ADDRESS SD_COLL
                        00000000 001F0          .LONG   0
                        00000000V 001F4         .ADDRESS SET_COLL
                        00000000 001F8          .LONG   0
                        00000000' 001FC         .ADDRESS SD_STAT
00000000  00000000  0000000B C0200              .LONG   11, 0, 0
                        00000000' 0020C         .ADDRESS SD_DUPL
00000000  00000000  0000000E 00210              .LONG   14, 0, 0
                        00000000' 0021C         .ADDRESS SD_SEQU
00000000  00000000  00000002 00220              .LONG   2, 0, 0
                        00000000' 0022C         .ADDRESS SD_RELA
00000000  00000000  00000003 00230              .LONG   3, 0, 0
                        00000000' 0023C         .ADDRESS SD_INDE
00000000  00000000  00000004 00240              .LONG   4, 0, 0
                        00000000' 0024C         .ADDRESS SD_BUCK
                        00000006 00250          .LONG   6
                        00000000V 00254         .ADDRESS SET_BUCK
                        00000000 00258          .LONG   0
                        00000000' 0025C         .ADDRESS SD_ALLO
                        00000007 00260          .LONG   7
                        00000000V 00264         .ADDRESS SET_ALLO
                        00000000 00268          .LONG   0
                        00000000' 0026C         .ADDRESS SD_CONT
00000000  00000000  00000009 00270              .LONG   9, 0, 0
```

```
                        00000000' 0027C         .ADDRESS SD_OVER                        ;
00000000  00000000      00000005  00280         .LONG    5, 0, 0                        ;
                        00000000' 0028C         .ADDRESS SD_LOAD                        ;
00000000  00000000      0000000C  00290         .LONG    12, 0, 0                       ;

                                                .PSECT   $OWN$,NOEXE,2

                                00000 STRING_DESC:
                                                .BLKB    8
                                00008 SUBVAL_DESC:
                                                .BLKB    8
                                00010 CMD_CLEAR0:
                                                .BLKB    0
                                00010 ORG:      .BLKB    1
                                00011 RFM:      .BLKB    1
                                00012 BKS:      .BLKB    1
                                00013           .BLKB    1
                                00014 BLS:      .BLKB    2
                                00016 MRS:      .BLKB    2
                                00018 OUT_ALQ:  .BLKB    4
                                0001C FOP:      .BLKB    4
                                00020 KBF:      .BLKB    4
                                00024 LRL:      .BLKB    2
                                00026           .BLKB    2
                                00028 INP_ALQ:  .BLKB    4
                                0002C WRK:      .BLKB    1
                                0002D TYP:      .BLKB    1
                                0002E           .BLKB    2
                                00030 OPT:      .BLKB    4
                                00034 CTX:      .BLKB    4
                                00038 FSZ:      .BLKB    1
                                00039           .BLKB    3
                                0003C FLAGS:    .BLKB    4
                                00040 CMD_CLEAR1:
                                                .BLKB    0

                                        SD_PROC=            P.AAA
                                        SD_WORK=            P.AAC
                                        SD_CHEC=            P.AAE
                                        SD_KEY=             P.AAG
                                        SD_SPEC=            P.AAI
                                        SD_STAB=            P.AAK
                                        SD_COLL=            P.AAM
                                        SD_STAT=            P.AAO
                                        SD_DUPL=            P.AAQ
                                        SD_SEQU=            P.AAS
                                        SD_RELA=            P.AAU
                                        SD_INDE=            P.AAW
                                        SD_BUCK=            P.AAY
                                        SD_ALLO=            P.ABA
                                        SD_CONT=            P.ABC
                                        SD_OVER=            P.ABE
                                        SD_LOAD=            P.ABG
                                        SD_P1=              P.ABI
                                        SD_P2=              P.ABK
                                        SD_$VER=            P.ABM
                                        SD_FORM=            P.ABO                       ;
```

```
                                        CLIQUALDESC=        P.ABQ
                                                .EXTRN  CLI$GET_VALUE, CLI$PRESENT
                                                .EXTRN  CLI$NEXT_QUAL, SOR$M_STABLE
                                                .EXTRN  SOR$M_SEQ_CHECK
                                                .EXTRN  SOR$M_NODUPS, SOR$M_EBCDIC
                                                .EXTRN  SOR$M_MULTI, SOR$M_LOAD_FILL
                                                .EXTRN  SOR$GK_RECORD, SOR$GK_TAG
                                                .EXTRN  SOR$GK_ADDRESS, SOR$GK_INDEX
                                                .EXTRN  STR$COPY_R, SOR$SPEC_FILE
                                                .EXTRN  SOR$PASS_FILES, SOR$BEGIN_SORT
                                                .EXTRN  SOR$BEGIN_MERGE
                                                .EXTRN  LIB$SIGNAL, CLI$_VALCNVERR
                                                .EXTRN  CLI$_IVCHAR, CLI$_IVVALU
                                                .EXTRN  CLI$_NEGATED, CLI$_INVQUAL

                                                .PSECT  $CODE$,NOWRT,2

                                    0000 00000 CMD_ERROR:
                                                .WORD   Save nothing                    ; 0616
    00000000G   00          6C  FA 00002        CALLG   (AP), LIB$SIGNAL                ; 0650
50      04   AC 10000000   8F  C9 00009        BISL3   #268435456, ERR, R0             ; 0651
                                    04 00012    RET                                     ; 0652

; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  486     0653  1 GLOBAL ROUTINE SOR$$COMMAND
;  487     0654  1         (
;  488     0655  1         CONTEXT:        REF VECTOR[1,LONG],
;  489     0656  1         SORT_FLAG:      REF VECTOR[1,BYTE],
;  490     0657  1         STATISTICS:     REF VECTOR[1,BYTE],
;  491     0658  1         IMAGE_AP:       REF $BBLOCK
;  492     0659  1         ) =
;  493     0660  1 !++
;  494     0661  1 ! Functional Description:
;  495     0662  1 !
;  496     0663  1 !       This routine gets the sort command line from DCL and calls DCL to
;  497     0664  1 !       parse it.
;  498     0665  1 !
;  499     0666  1 ! Formal Parameters:
;  500     0667  1 !
;  501     0668  1 !       CONTEXT         Address of pointer to the context area
;  502     0669  1 !       SORT_FLAG       Flag indicating we were invoked by SORT (not MERGE)
;  503     0670  1 !       STATISTICS      Address in which to return whether statistics requested
;  504     0671  1 !       IMAGE_AP        Address of main program's argument list
;  505     0672  1 !
;  506     0673  1 ! Implicit Inputs:
;  507     0674  1 !
;  508     0675  1 !       NONE
;  509     0676  1 !
;  510     0677  1 ! Implicit Outputs:
;  511     0678  1 !
;  512     0679  1 !       NONE
;  513     0680  1 !
;  514     0681  1 ! Routine Value:
;  515     0682  1 !
;  516     0683  1 !       Status value
;  517     0684  1 !
;  518     0685  1 ! Side Effects:
;  519     0686  1 !
;  520     0687  1 !       Calls PASS_FILES.
;  521     0688  1 !
;  522     0689  1 !--
;  523     0690  1
;  524     0691  2     BEGIN
;  525     0692  2     LOCAL
;  526     0693  2         KEY_BUFFER:     VECTOR[1+4*MAX_KEYS,WORD],
;  527     0694  2         OUT_FILE_DESC:  $BBLOCK[DSC$C_D_BLN],
;  528     0695  2         INP_FILE_DESC:  $BBLOCK[DSC$C_D_BLN],
;  529     0696  2         STATUS;
;  530     0697  2     MACRO
;  531   M 0698  2         LENADR_(X) = %IF %ISSTRING(X)
;  532   M 0699  2             %THEN %CHARCOUNT(X), UPLIT BYTE(X)
;  533     0700  2             %ELSE .X[DSC$W_LENGTH], .X[DSC$A_POINTER] %FI %,
;  534   M 0701  2         DSC_INIT_(X,CLASS) =
;  535   M 0702  2             BEGIN
;  536   M 0703  2             X[DSC$W_LENGTH] = 0;
;  537   M 0704  2             X[DSC$B_CLASS] = %NAME('DSC$K_CLASS_',CLASS);
;  538   M 0705  2             X[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  539   M 0706  2             X[DSC$A_POINTER] = 0;
;  540     0707  2             END %;
;  541     0708  2
;  542     0709  2     ! Initialize out dynamic string descriptors
```

```
;  543          0710  2      !
;  544          0711  2      DSC_INIT_(OUT_FILE_DESC, 'D');
;  545          0712  2      DSC_INIT_(INP_FILE_DESC, 'D');
;  546          0713  2      DSC_INIT_(STRING_DESC, 'D');
;  547          0714  2
;  548          0715  2      ! Clear our variables
;  549          0716  2      !
;  550          0717  2      CH$FILL(0, CMD_CLEAR1-CMD_CLEAR0, CMD_CLEAR0);
;  551          0718  2
;  552        L 0719  2  %IF FUN_K_CHECKPOINT
;  553        U 0720  2  %THEN
;  554        U 0721  2      ! If we're a checkpoint subprocess, call the CLI through a mailbox
;  555        U 0722  2      !
;  556        U 0723  2      CLI_BEGIN(IMAGE_AP[BASE_], CLI$PRESENT(SD_CHKP));
;  557          0724  2  %FI
;  558          0725  2
;  559          0726  2      ! Determine whether the user invoked SORT or MERGE
;  560          0727  2      ! Base this judgement on the verb from the command line.
;  561          0728  2      !
;  562          0729  2      CLI_GET_VALUE( SD_$VER, STRING_DESC[BASE_] );
;  563          0730  2      FLAGS[IOS_SORT] = SORT_FLAG[0] = TRUE;      ! Assume we were invoked by SORT
;  564          0731  2      IF CH$EQL(LENADR_(STRING_DESC), LENADR_('MERG'))    ! JUST FOUR CHARACTERS!
;  565          0732  2      THEN
;  566          0733  2          FLAGS[IOS_SORT] = SORT_FLAG[0] = FALSE;
;  567          0734  2
;  568          0735  2      ! Get the context longword
;  569          0736  2      !
;  570          0737  2      CTX = CONTEXT[0];
;  571          0738  2
;  572          0739  2
;  573          0740  2      ! Zero the number of keys, and indicate that each key is "not yet used",
;  574          0741  2      ! by putting -1 in the KEY_ORDER field.
;  575          0742  2      !
;  576          0743  2      KBF = KEY_BUFFER[0];
;  577          0744  2      KBF[KEY_NUMBER] = 0;                            ! No keys yet
;  578          0745  2      DECR I FROM MAX_KEYS-1 TO 0 DO
;  579          0746  3          BEGIN
;  580          0747  3          BLOCK[ KBF[KEY_KBF(.I)], KBF_ORDER_S] = -1;
;  581          0748  2          END;
;  582          0749  2
;  583          0750  2
;  584          0751  2      ! Get output file spec and its qualifiers
;  585          0752  2      !
;  586          0753  2      CLI_GET_VALUE(SD_P2, OUT_FILE_DESC[BASE_]);
;  587          0754  2      SET_OUT_FMT();
;  588          0755  2
;  589          0756  2      ! Process all the command qualifiers and output fille qualifiers
;  590          0757  2      !
;  591          0758  2      INCR I FROM 0 TO STR_K_QUALCODES-1 DO
;  592          0759  3          BEGIN
;  593          0760  3          LOCAL
;  594          0761  3              Q:  REF VECTOR;
;  595          0762  3          Q = CLIQUALDESC[4*.I];
;  596          0763  4          IF (STATUS = CLI_PRESENT(.Q[0]))
;  597          0764  3          THEN
;  598          0765  4              BEGIN
;  599          0766  4              FLAGS[.Q[1]] = TRUE;
```

```
;  600          0767  4              IF .Q[2] NEQ 0 THEN (.Q[2])();
;  601          0768  4              END
;  602          0769  3          ELSE
;  603          0770  4              BEGIN
;  604          0771  4              IF .Q[3] NEQ 0 THEN (.Q[3])();
;  605          0772  3              END;
;  606          0773  2          END;
;  607          0774  2
;  608          0775  2
;  609          0776  2      ! Get input file spec and its qualifiers
;  610          0777  2      !
;  611          0778  2      CLI_GET_VALUE(SD_P1, INP_FILE_DESC[BASE_]);
;  612          0779  2      SET_INP_FMT();
;  613          0780  2
;  614          0781  2
;  615          0782  2      ! Copy all output options specified to output file parameter
;  616          0783  2      ! area.
;  617          0784  2      !
;  618          0785  2      IF .FLAGS[IOS_OVER] THEN FOP = .FOP OR FAB$M_CIF;
;  619          0786  2      IF .FLAGS[IOS_CONT] THEN FOP = .FOP OR FAB$M_CTG;
;  620          0787  2      IF .FLAGS[IOS_SEQU] THEN ORG = FAB$C_SEQ ELSE
;  621          0788  2      IF .FLAGS[IOS_RELA] THEN ORG = FAB$C_REL ELSE
;  622          0789  2      IF .FLAGS[IOS_INDE] THEN ORG = FAB$C_IDX;
;  623          0790  2
;  624          0791  2      ! As the utility, always set the "output file parse" option.
;  625          0792  2      !
;  626          0793  2      FOP = .FOP OR FAB$M_OFP;
;  627          0794  2
;  628          0795  2
;  629          0796  2      ! Copy the filenames.
;  630          0797  2      !
;  631          0798  2      STATUS = SOR$PASS_FILES(
;  632          0799  2          INP_FILE_DESC[BASE_],           ! Descriptor for input file
;  633          0800  2          OUT_FILE_DESC[BASE_],           ! Descriptor for output file
;  634          0801  3          (IF (.FLAGS AND 1^IOS_SEQU+1^IOS_RELA+1^IOS_INDE) EQL 0
;  635          0802  2              THEN 0 ELSE ORG),                                   ! ORG
;  636          0803  2          (IF .RFM EQL 0 THEN 0 ELSE RFM),                        ! RFM
;  637          0804  2          (IF .FLAGS[IOS_BUCK] THEN BKS ELSE 0),                  ! BKS
;  638          0805  2          (IF .BLS EQL 0 THEN 0 ELSE BLS),                        ! BLS
;  639          0806  2          (IF .MRS EQL 0 THEN 0 ELSE MRS),                        ! MRS
;  640          0807  2          (IF .FLAGS[IOS_ALLO] THEN OUT_ALQ ELSE 0),              ! ALQ
;  641          0808  2          (IF .FOP EQL 0 THEN 0 ELSE FOP),                        ! FOP
;  642          0809  2          (IF .FSZ EQL 0 THEN 0 ELSE FSZ),                        ! FSZ
;  643          0810  2          CONTEXT[0]);                                            ! Context
;  644          0811  2      IF NOT .STATUS THEN RETURN .STATUS;
;  645          0812  2
;  646          0813  2
;  647          0814  2      ! Copy Options
;  648          0815  2      !
;  649          0816  2      IF .FLAGS[IOS_STAB] THEN OPT = .OPT OR SOR$M_STABLE;
;  650          0817  2      IF .FLAGS[IOS_CHEC] THEN OPT = .OPT OR SOR$M_SEQ_CHECK;
;  651          0818  2      IF .FLAGS[IOS_LOAD] THEN OPT = .OPT OR SOR$M_LOAD_FILL;
;  652          0819  2      IF NOT .FLAGS[IOS_DUPL] THEN OPT = .OPT OR SOR$M_NODUPS;
;  653          0820  2      STATISTICS[0] = .FLAGS[IOS_STAT];
;  654          0821  2
;  655          0822  2
;  656          0823  2      ! Now get the rest of the input files.
```

```
;  657          0824  2     !
;  658          0825  2     WHILE CLI_GET_VALUE(SD_P1, INP_FILE_DESC[BASE_]) DO
;  659          0826  3         BEGIN
;  660          0827  3         SET_INP_FMT();
;  661          0828  3
;  662          0829  3         STATUS = SOR$PASS_FILES(
;  663          0830  3             INP_FILE_DESC[BASE_],       ! Descriptor for input file
;  664          0831  3             0,                          ! Descriptor for output file
;  665          0832  3             0,0,0,0,                    ! ORG,RFM,BKS,BLS,
;  666          0833  3             0,0,0,0,                    ! MRS,ALQ,FOP,FSZ
;  667          0834  3             CONTEXT[0]);                ! Context
;  668          0835  3         IF NOT .STATUS THEN RETURN .STATUS;
;  669          0836  2         END;
;  670          0837  2
;  671        L 0838  2 %IF FUN_K_CHECKPOINT
;  672        U 0839  2 %THEN
;  673        U 0840  2     ! Indicate that no other CLI_xxx calls are needed
;  674        U 0841  2     !
;  675        U 0842  2     CLI_END();
;  676          0843  2 %FI
;  677          0844  2
;  678          0845  2     ! Set default options
;  679          0846  2     !
;  680          0847  2     !OPT = .OPT OR 0;                   ! No other defaults
;  681          0848  2
;  682          0849  2
;  683          0850  2     ! Push all the key descriptions together (this happens if the user
;  684          0851  2     ! uses NUMBER:n for non-successive n).
;  685          0852  2     !
;  686          0853  3     BEGIN
;  687          0854  3     LOCAL
;  688          0855  3         P: REF BLOCK,
;  689          0856  3         Q: REF BLOCK;
;  690          0857  3     P = KBF[KEY_KBF(0)];
;  691          0858  3     Q = KBF[KEY_KBF(0)];
;  692          0859  3     INCR I FROM 0 TO .KBF[KEY_NUMBER]-1 DO
;  693          0860  4         BEGIN
;  694          0861  4         IF .P[KBF_ORDER_S] GEQ 0                ! Do a signed fetch
;  695          0862  4         THEN
;  696          0863  5             BEGIN
;  697          0864  5             CH$MOVE(%UPVAL*KBF_S_ENTRY, P[BASE_], Q[BASE_]);
;  698          0865  5             Q = Q[BASE_] + %UPVAL*KBF_S_ENTRY;
;  699          0866  5             END
;  700          0867  4         ELSE
;  701          0868  4             KBF[KEY_NUMBER] = .KBF[KEY_NUMBER] - 1;
;  702          0869  4         P = P[BASE_] + %UPVAL*KBF_S_ENTRY;
;  703          0870  3         END;
;  704          0871  2     END;
;  705          0872  2
;  706          0873  2
;  707          0874  2     IF .FLAGS[IOS_SORT]
;  708          0875  2     THEN
;  709          0876  3         BEGIN
;  710          0877  3         !
;  711          0878  3         ! Check for specifying non-SORT options
;  712          0879  3         !
;  713          0880  3         IF .FLAGS[IOS_CHEC]
```

```
;  714          0881  3          THEN
;  715          0882  3              RETURN CMD_ERROR(CLI$_INVQUAL);
;  716          0883  3          !
;  717          0884  3          ! Call BEGIN_SORT
;  718          0885  3          !
;  719          0886  3          STATUS = SOR$BEGIN_SORT(
;  720          0887  4              (IF .FLAGS[IOS_KEY]
;  721          0888  3              THEN KBF[BASE_] ELSE 0),     ! Addr of key buffer
;  722          0889  3              LRL,                         ! Longest record length
;  723          0890  3              OPT,                         ! Options longword
;  724          0891  3              INP_ALQ,                     ! File allocation amount
;  725          0892  3              0,                           ! Comparison routine
;  726        U 0893  3 %IF FUN_K_KANJI %THEN
;  727        U 0894  3              IF .FLAGS[IOS_BOTH] THEN DEL_BOTH ELSE
;  728          0895  3 %FI
;  729          0896  3              0,                           ! No equal key routine
;  730          0897  4              (IF .FLAGS[IOS_PROC]
;  731          0898  3              THEN TYP ELSE 0),            ! Sort type
;  732          0899  4              (IF .FLAGS[IOS_WORK]
;  733          0900  3              THEN WRK ELSE 0),            ! Number of work files
;  734          0901  3              CONTEXT[0]);                 ! Context parameter
;  735          0902  3          END
;  736          0903  2      ELSE
;  737          0904  3          BEGIN
;  738          0905  3          !
;  739          0906  3          ! Check for specifying non-MERGE options
;  740          0907  3          !
;  741          0908  3          IF .FLAGS[IOS_WORK] OR .FLAGS[IOS_PROC]
;  742          0909  3          THEN
;  743          0910  3              RETURN CMD_ERROR(CLI$_INVQUAL);
;  744          0911  3          !
;  745          0912  3          ! Call BEGIN_MERGE
;  746          0913  3          !
;  747          0914  3          STATUS = SOR$BEGIN_MERGE(
;  748          0915  4              (IF .FLAGS[IOS_KEY]
;  749          0916  3              THEN KBF[BASE_] ELSE 0),     ! Addr of key buffer
;  750          0917  3              LRL,                         ! Longest record length
;  751          0918  3              OPT,                         ! Options longword
;  752          0919  3              0,                           ! Merge order
;  753          0920  3              0,                           ! Comparison routine
;  754        U 0921  3 %IF FUN_K_KANJI %THEN
;  755        U 0922  3              IF .FLAGS[IOS_BOTH] THEN DEL_BOTH ELSE
;  756          0923  3 %FI
;  757          0924  3              0,                           ! No equal key routine
;  758          0925  3              0,                           ! No input routine
;  759          0926  3              CONTEXT[0]);                 ! Context parameter
;  760          0927  2          END;
;  761          0928  2      IF NOT .STATUS THEN RETURN .STATUS;
;  762          0929  2
;  763          0930  2      RETURN SS$_NORMAL;
;  764          0931  1      END;


                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                            47  52  45  4D  0029C P.ABR:  .ASCII  \MERG\                                    ;
```

```
                                                        .PSECT  $CODE$,NOWRT,2

                                                 0FFC 00000     .ENTRY  SOR$$COMMAND, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 0653
                                                                        R10,R11
                                          5E     F7F8    CE  9E 00002    MOVAB   -2056(SP), SP
                                    04    AE 020E0000    8F  D0 00007    MOVL    #34471936, OUT_FILE_DESC                ; 0711
                                                 08     AE  D4 0000F    CLRL    OUT_FILE_DESC+4
                                              020E0000  8F  DD 00012    PUSHL   #34471936                               ; 0712
                                                 04     AE  D4 00018    CLRL    INP_FILE_DESC+4
                                    0000'     CF 020E0000  8F  D0 0001B    MOVL    #34471936, STRING_DESC                  ; 0713
                                                 0000'  CF  D4 00024    CLRL    STRING_DESC+4
                              30          00     6E         00  2C 00028    MOVC5   #0, (SP), #0, #48, CMD_CLEARO           ; 0717
                                                 0000'  CF     0002D
                                                 0000'  CF  9F 00030    PUSHAB  STRING_DESC                             ; 0729
                                                 0000'  CF  9F 00034    PUSHAB  SD_$VER
                                    0000G     CF         02  FB 00038    CALLS   #2, CLI_GET_VALUE
                                      08      BC         01  90 0003D    MOVB    #1, @SORT_FLAG                          ; 0730
                                    0000'     CF         02  88 00041    BISB2   #2, FLAGS
                              04          00    0000'     DF     0000'  CF  2D 00046    CMPC5   STRING_DESC, @STRING_DESC+4, #0, #4, P.ABR ; 0731
                                                                 0000'  CF     0004F
                                                            08  12 00052    BNEQ    1$
                                                         08     BC  94 00054    CLRB    @SORT_FLAG                      ; 0733
                                    0000'     CF         02  8A 00057    BICB2   #2, FLAGS
                                    0000'     CF      04     AC  D0 0005C 1$: MOVL    CONTEXT, CTX                    ; 0737
                                    0000'     CF      10     AE  9E 00062    MOVAB   KEY_BUFFER, KBF                 ; 0743
                                                 0000'  DF     B4 00068    CLRW    @KBF                            ; 0744
                                              50    FE     8F  9A 0006C    MOVZBL  #254, I                         ; 0745
                              51              50            01  78 00070 2$: ASHL    #1, I, R1                       ; 0747
                                              51     0000'DF41  DE 00074    MOVAL   @KBF[R1], R1
                                        04    A1            01  AE 0007A    MNEGW   #1, 4(R1)
                                              EF            50  F4 0007E    SOBGEQ  I, 2$                           ; 0745
                                                         08     AE  9F 00081    PUSHAB  OUT_FILE_DESC                   ; 0753
                                                 0000'  CF     9F 00084    PUSHAB  SD_P2
                                    0000G     CF         02  FB 00088    CALLS   #2, CLI_GET_VALUE
                                    0000V     CF         00  FB 0008D    CALLS   #0, SET_OUT_FMT                 ; 0754
                                                            52  D4 00092    CLRL    I                               ; 0758
                              50              52            02  78 00094 3$: ASHL    #2, I, R0                       ; 0762
                                              53     0000'CF40  DE 00098    MOVAL   CLIQUALDESC[R0], Q
                                                            63  DD 0009E    PUSHL   (Q)                             ; 0763
                                    0000G     CF         01  FB 000A0    CALLS   #1, CLI_PRESENT
                                              5B            50  D0 000A5    MOVL    R0, STATUS
                                              12            5B  E9 000A8    BLBC    STATUS, 5$
                              00    0000'     CF      04     A3  E2 000AB    BBSS    4(Q), FLAGS, 4$                 ; 0766
                                                         08     A3  D5 000B2 4$: TSTL    8(Q)                            ; 0767
                                                            0F  13 000B5    BEQL    6$
                                      08      B3         00  FB 000B7    CALLS   #0, @8(Q)
                                                            09  11 000BB    BRB     6$                              ; 0763
                                                         0C     A3  D5 000BD 5$: TSTL    12(Q)                           ; 0771
                                                            04  13 000C0    BEQL    6$
                                      0C      B3         00  FB 000C2    CALLS   #0, @12(Q)
                              CA              52            10  F3 000C6 6$: AOBLEQ  #16, I, 3$                     ; 0758
                                              5E            DD  000CA    PUSHL   SP                              ; 0778
                                                 0000'  CF     9F 000CC    PUSHAB  SD_P1
                                    0000G     CF         02  FB 000D0    CALLS   #2, CLI_GET_VALUE
```

```
                0000V  CF              00 FB 000D5          CALLS   #0, SET_INP_FMT                    ; 0779
          05    0000'  CF              05 E1 000DA          BBC     #5, FLAGS, 7$                      ; 0785
                0000'  CF              02 88 000E0          BISB2   #2, FOP+3
          05    0000'  CF              01 E1 000E5 7$:      BBC     #1, FLAGS+1, 8$                    ; 0786
                0000'  CF              10 88 000EB          BISB2   #16, FOP+2
          06    0000'  CF              02 E1 000F0 8$:      BBC     #2, FLAGS, 9$                      ; 0787
                             0000'  CF 94 000F6             CLRB    ORG
                                    18 11 000FA             BRB     11$
          07    0000'  CF              03 E1 000FC 9$:      BBC     #3, FLAGS, 10$                     ; 0788
                0000'  CF              10 90 00102          MOVB    #16, ORG
                                       0B 11 00107          BRB     11$
          05    0000'  CF              04 E1 00109 10$:     BBC     #4, FLAGS, 11$                     ; 0789
                0000'  CF              20 90 0010F          MOVB    #32, ORG
                0000'  CF              20 88 00114 11$:     BISB2   #32, FOP+3                         ; 0793
                                 04    AC DD 00119          PUSHL   CONTEXT                            ; 0810
                             0000'  CF 95 0011C             TSTB    FSZ                                ; 0809
                                    04 12 00120             BNEQ    12$
                                    7E D4 00122             CLRL    -(SP)
                                    07 11 00124             BRB     13$
                          50    0000'  CF 9E 00126 12$:     MOVAB   FSZ, R0
                                50 DD 0012B                 PUSHL   R0
                             0000'  CF D5 0012D 13$:        TSTL    FOP                                ; 0808
                                    04 12 00131             BNEQ    14$
                                    7E D4 00133             CLRL    -(SP)
                                    07 11 00135             BRB     15$
                          50    0000'  CF 9E 00137 14$:     MOVAB   FOP, R0
                                50 DD 0013C                 PUSHL   R0
                             0000'  CF 95 0013E 15$:        TSTB    FLAGS                              ; 0807
                                    09 18 00142             BGEQ    16$
                          50    0000'  CF 9E 00144          MOVAB   OUT_ALQ, R0
                                50 DD 00149                 PUSHL   R0
                                    02 11 0014B             BRB     17$
                                    7E D4 0014D 16$:        CLRL    -(SP)
                             0000'  CF B5 0014F 17$:        TSTW    MRS                                ; 0806
                                    04 12 00153             BNEQ    18$
                                    7E D4 00155             CLRL    -(SP)
                                    07 11 00157             BRB     19$
                          50    0000'  CF 9E 00159 18$:     MOVAB   MRS, R0
                                50 DD 0015E                 PUSHL   R0
                             0000'  CF B5 00160 19$:        TSTW    BLS                                ; 0805
                                    04 12 00164             BNEQ    20$
                                    7E D4 00166             CLRL    -(SP)
                                    07 11 00168             BRB     21$
                          50    0000'  CF 9E 0016A 20$:     MOVAB   BLS, R0
                                50 DD 0016F                 PUSHL   R0
          09    0000'  CF              06 E1 00171 21$:     BBC     #6, FLAGS, 22$                     ; 0804
                          50    0000'  CF 9E 00177          MOVAB   BKS, R0
                                50 DD 0017C                 PUSHL   R0
                                    02 11 0017E             BRB     23$
                                    7E D4 00180 22$:        CLRL    -(SP)
                             0000'  CF 95 00182 23$:        TSTB    RFM                                ; 0803
                                    04 12 00186             BNEQ    24$
                                    7E D4 00188             CLRL    -(SP)
                                    07 11 0018A             BRB     25$
                          50    0000'  CF 9E 0018C 24$:     MOVAB   RFM, R0
                                50 DD 00191                 PUSHL   R0
                          1C    0000'  CF 93 00193 25$:     BITB    FLAGS, #28                         ; 0801
```

```
                                                04  12 00198          BNEQ    26$                                     ;
                                                7E  D4 0019A          CLRL    -(SP)                                   ;
                                                07  11 0019C          BRB     27$                                     ;
                                     50   0000' CF  9E 0019E 26$:     MOVAB   ORG, R0                                 ;
                                                50  DD 001A3          PUSHL   R0                                      ;
                                       2C       AE  9F 001A5 27$:     PUSHAB  OUT_FILE_DESC                           ; 0800
                                       28       AE  9F 001A8          PUSHAB  INP_FILE_DESC                           ; 0799
                              00000000G 00      FB  01 001AB          CALLS   #11, SOR$PASS_FILES                     ; 0810
                                        5B      50  D0 001B2          MOVL    R0, STATUS                              ;
                                        77      5B  E9 001B5          BLBC    STATUS, 33$                             ; 0811
                                  09    0000' CF  02  E1 001B8          BBC     #2, FLAGS+1, 28$                        ; 0816
                                        0000' CF  8F  C8 001BE          BISL2   #SOR$M_STABLE, OPT                      ;
                                  09    0000' CF  05  E1 001C7 28$:     BBC     #5, FLAGS+1, 29$                        ; 0817
                                        0000' CF  8F  C8 001CD          BISL2   #SOR$M_SEQ_CHECK, OPT                   ;
                                  09    0000' CF  04  E1 001D6 29$:     BBC     #4, FLAGS+1, 30$                        ; 0818
                                        0000' CF  8F  C8 001DC          BISL2   #SOR$M_LOAD_FILL, OPT                   ;
                                  09    0000' CF  06  E0 001E5 30$:     BBS     #6, FLAGS+1, 31$                        ; 0819
                                        0000' CF  8F  C8 001EB          BISL2   #SOR$M_NODUPS, OPT                      ;
                        50  0000' CF        01  03  EF 001F4 31$:     EXTZV   #3, #1, FLAGS+1, R0                     ; 0820
                                    0C      BC  50  90 001FB          MOVB    R0, @STATISTICS                         ;
                                                5E  DD 001FF 32$:     PUSHL   SP                                      ; 0825
                                       0000' CF  9F 00201          PUSHAB  SD_P1                                   ;
                                     0000G CF  02  FB 00205          CALLS   #2, CLI_GET_VALUE                       ;
                                        25      50  E9 0020A          BLBC    R0, 34$                                 ;
                                     0000V CF  00  FB 0020D          CALLS   #0, SET_INP_FMT                         ; 0827
                                       04       AC  DD 00212          PUSHL   CONTEXT                                 ; 0834
                                                7E  7C 00215          CLRQ    -(SP)                                   ;
                                                7E  7C 00217          CLRQ    -(SP)                                   ;
                                                7E  7C 00219          CLRQ    -(SP)                                   ;
                                                7E  7C 0021B          CLRQ    -(SP)                                   ;
                                                7E  D4 0021D          CLRL    -(SP)                                   ;
                                       28       AE  9F 0021F          PUSHAB  INP_FILE_DESC                           ; 0830
                              00000000G 00      FB  01 00222          CALLS   #11, SOR$PASS_FILES                     ; 0834
                                        5B      50  D0 00229          MOVL    R0, STATUS                              ;
                                        D0      5B  E8 0022C          BLBS    STATUS, 32$                             ; 0835
                                              00C2  31 0022F 33$:     BRW     51$                                     ;
                                        56    0000' CF  D0 00232 34$:     MOVL    KBF, R6                             ; 0857
                                        50      02  A6  9E 00237          MOVAB   2(R6), R0                           ;
                                        57          50  D0 0023B          MOVL    R0, P                               ;
                                        59          50  D0 0023E          MOVL    R0, Q                               ; 0858
                                        5A          66  3C 00241          MOVZWL  (R6), R10                           ; 0859
                                        58          01  CE 00244          MNEGL   #1, I                               ;
                                                    13  11 00247          BRB     38$                                 ;
                                       02       A7  B5 00249 35$:     TSTW    2(P)                                    ; 0861
                                                09  19 0024C          BLSS    36$                                     ;
                    69                  67      08  28 0024E          MOVC3   #8, (P), (Q)                            ; 0864
                                        59          08  C0 00252          ADDL2   #8, Q                               ; 0865
                                                    02  11 00255          BRB     37$                                 ; 0861
                                                    66  B7 00257 36$:     DECW    (R6)                                ; 0868
                                        57          08  C0 00259 37$:     ADDL2   #8, P                               ; 0869
                    E9                  58          5A  F2 0025C 38$:     AOBLSS  R10, I, 35$                         ; 0859
                    4E          0000' CF            01  E1 00260          BBC     #1, FLAGS, 45$                      ; 0874
                    52          0000' CF            05  E0 00266          BBS     #5, FLAGS+1, 46$                    ; 0880
                                       04       AC  DD 0026C          PUSHL   CONTEXT                                 ; 0901
                                        09    0000' CF  E9 0026F          BLBC    FLAGS+1, 39$                        ; 0899
                                        50    0000' CF  9E 00274          MOVAB   WRK, R0                             ;
                                                    50  DD 00279          PUSHL   R0                                  ;
```

```
                        02  11 0027B          BRB     40$                                     ;
                        7E  D4 0027D 39$:      CLRL    -(SP)                                   ;
                  09    CF  E9 0027F 40$:      BLBC    FLAGS+2, 41$                            ; 0897
                  50    CF  9E 00284           MOVAB   TYP, R0                                 ;
                        50  DD 00289           PUSHL   R0                                      ;
                        02  11 0028B           BRB     42$                                     ;
                        7E  D4 0028D 41$:      CLRL    -(SP)                                   ;
                        7E  7C 0028F 42$:      CLRQ    -(SP)                                   ; 0901
                  0000' CF  9F 00291           PUSHAB  INP_ALQ                                 ; 0886
                  0000' CF  9F 00295           PUSHAB  OPT                                     ;
                  0000' CF  9F 00299           PUSHAB  LRL                                     ;
                  0000' CF  95 0029D           TSTB    FLAGS+1                                 ; 0887
                        06  18 002A1           BGEQ    43$                                     ;
                  0000' CF  DD 002A3           PUSHL   KBF                                     ; 0888
                        02  11 002A7           BRB     44$                                     ;
                        7E  D4 002A9 43$:      CLRL    -(SP)                                   ;
00000000G         00        09 FB 002AB 44$:   CALLS   #9, SOR$BEGIN_SORT                      ; 0901
                        3A  11 002B2           BRB     50$                                     ;
                  05    0000' CF  E8 002B4 45$:      BLBS    FLAGS+1, 46$                    ; 0908
                  0C    0000' CF  E9 002B9           BLBC    FLAGS+2, 47$                    ;
                  000000000G 8F   DD 002BE 46$:      PUSHL   #CLI$_INVQUAL                   ; 0910
              FD24  CF              01 FB 002C4           CALLS   #1, CMD_ERROR               ;
                                    04 002C9           RET                                     ;
                    04    AC  DD 002CA 47$:      PUSHL   CONTEXT                             ; 0926
                        7E  7C 002CD           CLRQ    -(SP)                                   ;
                        7E  7C 002CF           CLRQ    -(SP)                                   ;
                  0000' CF  9F 002D1           PUSHAB  OPT                                     ; 0914
                  0000' CF  9F 002D5           PUSHAB  LRL                                     ;
                  0000' CF  95 002D9           TSTB    FLAGS+1                                 ; 0915
                        06  18 002DD           BGEQ    48$                                     ;
                  0000' CF  DD 002DF           PUSHL   KBF                                     ; 0916
                        02  11 002E3           BRB     49$                                     ;
                        7E  D4 002E5 48$:      CLRL    -(SP)                                   ;
00000000G         00        08 FB 002E7 49$:   CALLS   #8, SOR$BEGIN_MERGE                     ; 0926
                  5B    50  D0 002EE 50$:      MOVL    R0, STATUS                              ;
                  04    5B  E8 002F1           BLBS    STATUS, 52$                             ; 0928
                  50    5B  D0 002F4 51$:      MOVL    STATUS, R0                              ;
                              04 002F7           RET                                         ;
                  50        01  D0 002F8 52$:      MOVL    #1, R0                              ; 0930
                                04 002FB           RET                                         ; 0931

; Routine Size:  764 bytes,    Routine Base:  $CODE$ + 0013
```

```
;  766     0932  1 ROUTINE UTIL$NUMERVAL
;  767     0933  1         (
;  768     0934  1         DESC:           REF $BBLOCK[DSC$C_S_BLN]
;  769     0935  1         ) =
;  770     0936  1 !++
;  771     0937  1 ! Functional Description:
;  772     0938  1 !
;  773     0939  1 !       This routine converts a string to a number.
;  774     0940  1 !       It allows for unary minus, and decimal, hexadecimal and octal radices.
;  775     0941  1 !
;  776     0942  1 ! Formal Parameters:
;  777     0943  1 !
;  778     0944  1 !       DESC            String descriptor
;  779     0945  1 !
;  780     0946  1 ! Implicit Inputs:
;  781     0947  1 !
;  782     0948  1 !       NONE
;  783     0949  1 !
;  784     0950  1 ! Implicit Outputs:
;  785     0951  1 !
;  786     0952  1 !       NONE
;  787     0953  1 !
;  788     0954  1 ! Routine Value:
;  789     0955  1 !
;  790     0956  1 !       Integer value of the converted string
;  791     0957  1 !
;  792     0958  1 ! Side Effects:
;  793     0959  1 !
;  794     0960  1 !       Signals CLI$_IVCHAR for invalid characters
;  795     0961  1 !
;  796     0962  1 !--
;  797     0963  2     BEGIN
;  798     0964  2     LOCAL
;  799     0965  2         C,          ! Current character
;  800     0966  2         V,          ! The computed value
;  801     0967  2         L,          ! Length of string
;  802     0968  2         A,          ! Address of string
;  803     0969  2         B,          ! Base
;  804     0970  2         N;          ! Negation flag
;  805     0971  2     MACRO
;  806     0972  2         ERR_(X) = (CMD_ERROR(X); RETURN 0) %;
;  807     0973  2
;  808     0974  2     L = .DESC[DSC$W_LENGTH];
;  809     0975  2     A = .DESC[DSC$A_POINTER];
;  810     0976  2     V = 0;
;  811     0977  2     B = 10;
;  812     0978  2     N = 1;
;  813     0979  2     WHILE (L = .L-1) GEQ 0 DO
;  814     0980  3         BEGIN
;  815     0981  3         SELECTONE (C = CH$RCHAR_A(A)) OF
;  816     0982  3             SET
;  817     0983  3             ['0' TO '9']:
;  818     0984  4                 BEGIN
;  819     0985  4                 V = .C - '0';
;  820     0986  4                 EXITLOOP;
;  821     0987  3                 END;
;  822     0988  3             ['%']:
```

```
;  823          0989  4                BEGIN
;  824          0990  4                IF (L=.L-1) GEQ 0
;  825          0991  4                THEN
;  826          0992  5                    BEGIN
;  827          0993  5                    SELECTONE (C = CH$RCHAR_A(A)) OF
;  828          0994  5                        SET
;  829          0995  5                        ['D']: B =    10 ;
;  830          0996  5                        ['O']: B = %O'10';
;  831          0997  5                        ['X']: B = %X'10';
;  832          0998  5                        [OTHERWISE]: ERR_(CLI$_IVCHAR);
;  833          0999  5                        TES
;  834          1000  5                    END
;  835          1001  4                ELSE
;  836          1002  4                    ERR_(CLI$_IVCHAR);
;  837          1003  4                EXITLOOP;
;  838          1004  3                END;
;  839          1005  3            ['-']:
;  840          1006  4                BEGIN
;  841          1007  4                N = -.N;
;  842          1008  4                IF .N GEQ 0 THEN ERR_(CLI$_IVCHAR);
;  843          1009  3                END;
;  844          1010  3            [OTHERWISE]:
;  845          1011  3                ERR_(CLI$_IVCHAR);
;  846          1012  3            TES;
;  847          1013  2        END;
;  848          1014  2
;  849          1015  2    WHILE (L = .L-1) GEQ 0 DO
;  850          1016  3        BEGIN
;  851          1017  3        SELECTONE (C = CH$RCHAR_A(A)) OF
;  852          1018  3            SET
;  853          1019  3            ['0' TO '9']: C = .C - '0';
;  854          1020  3            ['A' TO 'F']: C = .C - 'A' + 10;
;  855          1021  3            [OTHERWISE]: ERR_(CLI$_IVCHAR);
;  856          1022  3            TES;
;  857          1023  3        IF .C GEQU .B THEN ERR_(CLI$_IVCHAR);
;  858          1024  3        V = .V * .B + .C;
;  859          1025  2        END;
;  860          1026  2
;  861          1027  2    RETURN .V * .N;
;  862          1028  2
;  863          1029  1    END;


                                00FC 00000 UTIL$NUMERVAL:
                                                .WORD   Save R2,R3,R4,R5,R6,R7                  ; 0932
                    50    04    AC  D0 00002          MOVL    DESC, R0                          ; 0974
                    54          60  3C 00006          MOVZWL  (R0), L                           ;
                    57    04    A0  D0 00009          MOVL    4(R0), A                          ; 0975
                                53  D4 0000D          CLRL    V                                 ; 0976
                    55          0A  D0 0000F          MOVL    #10, B                            ; 0977
                    56          01  D0 00012          MOVL    #1, N                             ; 0978
                                54  D7 00015 1$:       DECL    L                                 ; 0979
                                55  19 00017          BLSS    6$                                ;
                    52          87  9A 00019          MOVZBL  (A)+, C                           ; 0981
```

```
                        30              52  D1 0001C          CMPL    C, #48                          ; 0983
                                        0B  19 0001F          BLSS    2$
                        39              52  D1 00021          CMPL    C, #57
                                        06  14 00024          BGTR    2$
                        53      D0      A2  9E 00026          MOVAB   -48(R2), V                      ; 0985
                                        42  11 0002A          BRB     6$                              ; 0984
                        25              52  D1 0002C 2$:      CMPL    C, #37                          ; 0988
                                        31  12 0002F          BNEQ    5$
                                        54  D7 00031          DECL    L                               ; 0990
                                        69  19 00033          BLSS    9$
                        52              87  9A 00035          MOVZBL  (A)+, C                         ; 0993
                00000044 8F             52  D1 00038          CMPL    C, #68                          ; 0995
                                        05  12 0003F          BNEQ    3$
                        55              0A  D0 00041          MOVL    #10, B
                                        28  11 00044          BRB     6$
                0000004F 8F             52  D1 00046 3$:      CMPL    C, #79                          ; 0996
                                        05  12 0004D          BNEQ    4$
                        55              08  D0 0004F          MOVL    #8, B
                                        1A  11 00052          BRB     6$
                00000058 8F             52  D1 00054 4$:      CMPL    C, #88                          ; 0997
                                        41  12 0005B          BNEQ    9$
                        55              10  D0 0005D          MOVL    #16, B
                                        0C  11 00060          BRB     6$
                        2D              52  D1 00062 5$:      CMPL    C, #45                          ; 1005
                                        37  12 00065          BNEQ    9$
                        56              56  CE 00067          MNEGL   N, N                            ; 1007
                                        A9  19 0006A          BLSS    1$                              ; 1008
                                        30  11 0006C          BRB     9$
                                        54  D7 0006E 6$:      DECL    L                               ; 1015
                                        43  19 00070          BLSS    11$
                            52          87  9A 00072          MOVZBL  (A)+, C                         ; 1017
                            30          52  D1 00075          CMPL    C, #48                          ; 1019
                                        0A  19 00078          BLSS    7$
                            39          52  D1 0007A          CMPL    C, #57
                                        05  14 0007D          BGTR    7$
                            52          30  C2 0007F          SUBL2   #48, C
                                        15  11 00082          BRB     8$
                        00000041 8F     52  D1 00084 7$:      CMPL    C, #65                          ; 1020
                                        11  19 0008B          BLSS    9$
                        00000046 8F     52  D1 0008D          CMPL    C, #70
                                        08  14 00094          BGTR    9$
                            52          37  C2 00096          SUBL2   #55, C
                            55          52  D1 00099 8$:      CMPL    C, B                            ; 1023
                                        0D  1F 0009C          BLSSU   10$
                                00000000G 8F    DD 0009E 9$:  PUSHL   #CLI$_IVCHAR
                            FC48    CF      01  FB 000A4          CALLS   #1, CMD_ERROR
                                        11  11 000A9          BRB     12$
                50              53      55  C5 000AB 10$:     MULL3   B, V, R0                        ; 1024
                53              50      52  C1 000AF          ADDL3   C, R0, V
                                        B9  11 000B3          BRB     6$                              ; 1015
                            53          56  C4 000B5 11$:     MULL2   N, R3                           ; 1027
                            50          53  D0 000B8          MOVL    R3, R0
                                            04 000BB          RET
                                        50  D4 000BC 12$:     CLRL    R0                              ; 1029
                                            04 000BE          RET

; Routine Size:  191 bytes,    Routine Base:  $CODE$ + 030F
```

```
;  865          1030  1 ROUTINE KEYVALU =
;  866          1031  1
;  867          1032  1 !++
;  868          1033  1 ! Functional Description:
;  869          1034  1 !
;  870          1035  1 !       This routine breaks a string into a keyword and a keyword value.
;  871          1036  1 !
;  872          1037  1 ! Formal Parameters:
;  873          1038  1 !
;  874          1039  1 !       NONE
;  875          1040  1 !
;  876          1041  1 ! Implicit Inputs:
;  877          1042  1 !
;  878          1043  1 !       STRING_DESC     String descriptor of source
;  879          1044  1 !
;  880          1045  1 ! Implicit Outputs:
;  881          1046  1 !
;  882          1047  1 !       SUBVAL_DESC     String descriptor for storing value
;  883          1048  1 !
;  884          1049  1 ! Routine Value:
;  885          1050  1 !
;  886          1051  1 !       True iff the source string contained a value.
;  887          1052  1 !
;  888          1053  1 ! Side Effects:
;  889          1054  1 !
;  890          1055  1 !       Signals CLI$_IVVALU for zero-length values
;  891          1056  1 !
;  892          1057  1 !--
;  893          1058  2     BEGIN
;  894          1059  2     BUILTIN
;  895          1060  2         LOCC,
;  896          1061  2         NULLPARAMETER;
;  897          1062  2     LOCAL
;  898          1063  2         L: WORD;
;  899          1064  2
;  900          1065  2     L = .STRING_DESC[DSC$W_LENGTH];
;  901          1066  2
;  902          1067  3     BEGIN REGISTER R0 = 0, R1 = 1;
;  903          1068  3     LOCC( %REF(':'), L, .STRING_DESC[DSC$A_POINTER]; R0, R1 );
;  904          1069  3     L = .L - .R0;
;  905          1070  3     LOCC( %REF('='), L, .STRING_DESC[DSC$A_POINTER]; R0, R1 );
;  906          1071  3     L = .L - .R0;
;  907          1072  2     END;
;  908          1073  2
;  909          1074  2     SUBVAL_DESC[DSC$W_LENGTH] = 0;      ! Indicate no value seen yet
;  910          1075  2
;  911          1076  2     IF .L GEQU .STRING_DESC[DSC$W_LENGTH] THEN RETURN FALSE;
;  912          1077  2     L = .L + 1;                         ! Remove the colon or equals
;  913          1078  2
;  914          1079  2     SUBVAL_DESC[DSC$A_POINTER] = .STRING_DESC[DSC$A_POINTER] + .L;
;  915          1080  2     SUBVAL_DESC[DSC$W_LENGTH]  = .STRING_DESC[DSC$W_LENGTH]  - .L;
;  916          1081  2
;  917          1082  2     IF .SUBVAL_DESC[DSC$W_LENGTH] EQL 0 THEN CMD_ERROR(CLI$_IVVALU);
;  918          1083  2
;  919          1084  2     RETURN TRUE;
;  920          1085  2
;  921          1086  1     END;
```

```
                                 003C 00000 KEYVALU::.WORD   Save R2,R3,R4,R5                        ; 1030
                          55      0000' CF   9E 00002        MOVAB    SUBVAL_DESC, R5
                          54        F8  A5   3C 00007        MOVZWL   STRING_DESC, R4                ; 1065
                          52            54   B0 0000B        MOVW     R4, L
                          53        FC  A5   D0 0000E        MOVL     STRING_DESC+4, R3              ; 1068
                    63    52            3A   3A 00012        LOCC     #58, L, (R3)
                          52            50   A2 00016        SUBW2    R0, L                          ; 1069
                    63    52            3D   3A 00019        LOCC     #61, L, (R3)                   ; 1070
                          52            50   A2 0001D        SUBW2    R0, L                          ; 1071
                                        65   B4 00020        CLRW     SUBVAL_DESC                    ; 1074
                          54            52   B1 00022        CMPW     L, R4                          ; 1076
                                        1F   1E 00025        BGEQU    2$
                                        52   B6 00027        INCW     L                              ; 1077
                          50            52   3C 00029        MOVZWL   L, R0                          ; 1079
                04  A5    53            50   C1 0002C        ADDL3    R0, R3, SUBVAL_DESC+4
                    65    54            52   A3 00031        SUBW3    L, R4, SUBVAL_DESC             ; 1080
                                        0B   12 00035        BNEQ     1$                             ; 1082
                             00000000G  8F   DD 00037        PUSHL    #CLI$_IVVALU
                      FBF0  CF         01   FB 0003D        CALLS    #1, CMD_ERROR
                          50            01   D0 00042 1$:    MOVL     #1, R0                         ; 1084
                                             04 00045        RET
                                        50   D4 00046 2$:    CLRL     R0                             ; 1086
                                             04 00048        RET                                     ;

; Routine Size:  73 bytes,    Routine Base:  $CODE$ + 03CE
```

```
;  923          1087  1 ROUTINE NUMERIC_PARSE
;  924          1088  1         (
;  925          1089  1         DESC:   REF $BBLOCK[DSC$C_D_BLN]
;  926          1090  1         ) =
;  927          1091  1
;  928          1092  1 !++
;  929          1093  1 ! Functional Description:
;  930          1094  1 !
;  931          1095  1 !       This routine converts an ASCII number into a binary value.
;  932          1096  1 !
;  933          1097  1 ! Formal Parameters:
;  934          1098  1 !
;  935          1099  1 !       DESC            The address of the ASCII string descriptor.
;  936          1100  1 !
;  937          1101  1 ! Implicit Inputs:
;  938          1102  1 !
;  939          1103  1 !       STRING_DESC     Used for issuing the "'XXX' keyword requires a value"
;  940          1104  1 !                       diagnostic.
;  941          1105  1 !
;  942          1106  1 ! Implicit Outputs:
;  943          1107  1 !
;  944          1108  1 !       None.
;  945          1109  1 !
;  946          1110  1 ! Routine Value:
;  947          1111  1 !
;  948          1112  1 !       Converted value
;  949          1113  1 !
;  950          1114  1 ! Side Effects:
;  951          1115  1 !
;  952          1116  1 !       None.
;  953          1117  1 !
;  954          1118  1 !--
;  955          1119  1
;  956          1120  2     BEGIN
;  957          1121  2     BUILTIN
;  958          1122  2         NULLPARAMETER;
;  959          1123  2
;  960          1124  2     IF .DESC[DSC$W_LENGTH] EQL 0
;  961          1125  2     THEN                                    ! No value where one was expected
;  962          1126  3         BEGIN
;  963          1127  3         CMD_ERROR(SOR$_SHR_NOVALUE, 1, STRING_DESC[BASE_]);
;  964          1128  3         RETURN 0;
;  965          1129  2         END;
;  966          1130  2
;  967          1131  2     ! Parse the value
;  968          1132  2     !
;  969          1133  2     RETURN UTIL$NUMERVAL(DESC[BASE_]);
;  970          1134  2
;  971          1135  1     END;


                        0000 00000 NUMERIC_PARSE:
                                           .WORD   Save nothing                    ; 1087
                04   BC  B5 00002          TSTW    @DESC                           ; 1124
```

```
                                 13  12 00005          BNEQ    1$                                          ;
                          0000'  CF  9F 00007          PUSHAB  STRING_DESC                         1127    ;
                                 01  DD 0000B          PUSHL   #1                                          ;
                      001C1104   8F  DD 0000D          PUSHL   #1839364                                    ;
                FBD1  CF         03  FB 00013          CALLS   #3, CMD_ERROR                               ;
                                 09  11 00018          BRB     2$                                  1128    ;
                             04  AC  DD 0001A 1$:       PUSHL   DESC                                1133    ;
                FED6  CF         01  FB 0001D          CALLS   #1, UTIL$NUMERVAL                           ;
                                     04 00022          RET                                                 ;
                                 50  D4 00023 2$:       CLRL    R0                                  1135    ;
                                     04 00025          RET                                                 ;

; Routine Size:  38 bytes,    Routine Base:  $CODE$ + 0417
```

```
;  973          1136  1 ROUTINE SET_SPEC: NOVALUE =
;  974          1137  1
;  975          1138  1 !++
;  976          1139  1 ! Functional Description:
;  977          1140  1 !
;  978          1141  1 !       This routine calls the CLI to parse a file name and stores it in the
;  979          1142  1 !       sort parameter area.
;  980          1143  1 !
;  981          1144  1 ! Formal Parameters:
;  982          1145  1 !
;  983          1146  1 !       None.
;  984          1147  1 !
;  985          1148  1 ! Implicit Inputs:
;  986          1149  1 !
;  987          1150  1 !       None.
;  988          1151  1 !
;  989          1152  1 ! Implicit Outputs:
;  990          1153  1 !
;  991          1154  1 !       None.
;  992          1155  1 !
;  993          1156  1 ! Routine Value:
;  994          1157  1 !
;  995          1158  1 !
;  996          1159  1 ! Completion Codes:
;  997          1160  1 !
;  998          1161  1 !       None.
;  999          1162  1 !
; 1000          1163  1 ! Side Effects:
; 1001          1164  1 !
; 1002          1165  1 !       None.
; 1003          1166  1 !
; 1004          1167  1 !--
; 1005          1168  1
; 1006          1169  2     BEGIN
; 1007          1170  2     LOCAL
; 1008          1171  2         P:      REF $BBLOCK,
; 1009          1172  2         STATUS;
; 1010          1173  2
; 1011          1174  2     P = STRING_DESC[BASE_];
; 1012          1175  2     IF NOT CLI_GET_VALUE( SD_SPEC, P[BASE_] )
; 1013          1176  2     THEN
; 1014          1177  2         P = $DESCRIPTOR(STR_DEF_SPECFILE);
; 1015          1178  2
; 1016          1179  2     STATUS = SOR$SPEC_FILE(P[BASE_], 0, .CTX);
; 1017          1180  2
; 1018          1181  2     IF NOT .STATUS THEN RETURN CMD_ERROR(.STATUS);
; 1019          1182  1     END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                    54  55  50  4E  49  24  53  59  53  002A0 P.ABT:  .ASCII  \SYS$INPUT\                          ;
                                                        002A9        .BLKB   3
                                            00000009    002AC P.ABS:  .LONG   9                                   ;
                                            00000000'   002B0        .ADDRESS P.ABT                             ;
```

```
                                                .PSECT  $CODE$,NOWRT,2

                                        0004 00000 SET_SPEC:
                                                .WORD   Save R2                                 ; 1136
                    52      0000'   CF  9E 00002       MOVAB   STRING_DESC, P                  ; 1174
                                    52  DD 00007       PUSHL   P                               ; 1175
                            0000'   CF  9F 00009       PUSHAB  SD_SPEC                         ;
                0000G   CF              02  FB 0000D   CALLS   #2, CLI_GET_VALUE               ;
                        05              50  E8 00012   BLBS    R0, 1$                        ;
                        52      0000'   CF  9E 00015   MOVAB   P.ABS, P                       ; 1177
                                0000'   CF  DD 0001A 1$:       PUSHL   CTX                     ; 1179
                                        7E  D4 0001E   CLRL    -(SP)                         ;
                                        52  DD 00020   PUSHL   P                               ;
            00000000G   00              03  FB 00022   CALLS   #3, SOR$SPEC_FILE              ;
                        07              50  E8 00029   BLBS    STATUS, 2$                    ; 1181
                                        50  DD 0002C   PUSHL   STATUS                        ;
                FB90    CF              01  FB 0002E   CALLS   #1, CMD_ERROR                 ;
                                            04 00033 2$:       RET                             ; 1182

; Routine Size:  52 bytes,    Routine Base:  $CODE$ + 043D
```

```
;  1021        U 1183  1 %IF %VARIANT %THEN
;  1022        U 1184  1 ROUTINE DEL_BOTH = (EXTERNAL LITERAL SOR$_DELBOTH; SOR$_DELBOTH);
;  1023        U 1185  1 ROUTINE SET_DUPL: NOVALUE =
;  1024        U 1186  1
;  1025        U 1187  1 !++
;  1026        U 1188  1 ! Functional Description:
;  1027        U 1189  1 !
;  1028        U 1190  1 !       This routine calls the CLI to parse a file name and stores it in the
;  1029        U 1191  1 !       sort parameter area.
;  1030        U 1192  1 !
;  1031        U 1193  1 ! Formal Parameters:
;  1032        U 1194  1 !
;  1033        U 1195  1 !       The address of the qualifier descriptor block.
;  1034        U 1196  1 !
;  1035        U 1197  1 !
;  1036        U 1198  1 ! Implicit Inputs:
;  1037        U 1199  1 !
;  1038        U 1200  1 !       None.
;  1039        U 1201  1 !
;  1040        U 1202  1 ! Implicit Outputs:
;  1041        U 1203  1 !
;  1042        U 1204  1 !       None.
;  1043        U 1205  1 !
;  1044        U 1206  1 ! Routine Value:
;  1045        U 1207  1 !
;  1046        U 1208  1 !
;  1047        U 1209  1 ! Completion Codes:
;  1048        U 1210  1 !
;  1049        U 1211  1 !       None.
;  1050        U 1212  1 !
;  1051        U 1213  1 ! Side Effects:
;  1052        U 1214  1 !
;  1053        U 1215  1 !       None.
;  1054        U 1216  1 !
;  1055        U 1217  1 !--
;  1056        U 1218  1
;  1057        U 1219  1     BEGIN
;  1058        U 1220  1     KEY_TABLE_('DELB', 'DELBOTH');
;  1059        U 1221  1     LOCAL
;  1060        U 1222  1         LOOKUP;
;  1061        U 1223  1
;  1062        U 1224  1     ! Name given?
;  1063        U 1225  1     !
;  1064        U 1226  1     IF NOT CLI_GET_VALUE( SD_DUPL, STRING_DESC[BASE_] ) THEN RETURN;
;  1065        U 1227  1
;  1066        U 1228  1     FLAGS[IOS_DUPL] = TRUE;
;  1067        U 1229  1
;  1068        U 1230  1     LOOKUP = LOOKUP_KEY(KEYWORD_TABLE);
;  1069        U 1231  1     IF .LOOKUP LSSU TABLE_SIZE THEN FLAGS[IOS_BOTH] = TRUE;
;  1070        U 1232  1
;  1071        U 1233  1     END;
;  1072          1234  1 %FI
```

```
;  1074          1235  1 ROUTINE SET_OUT_FMT: NOVALUE =
;  1075          1236  1
;  1076          1237  1 !++
;  1077          1238  1 ! Functional Description:
;  1078          1239  1 !
;  1079          1240  1 !       This routine sets the output file record format in the output file
;  1080          1241  1 !       parameter area.
;  1081          1242  1 !
;  1082          1243  1 ! Formal Parameters:
;  1083          1244  1 !
;  1084          1245  1 !       The address of the qualifier descriptor block.
;  1085          1246  1 !
;  1086          1247  1 ! Implicit Inputs:
;  1087          1248  1 !
;  1088          1249  1 !       None.
;  1089          1250  1 !
;  1090          1251  1 ! Implicit Outputs:
;  1091          1252  1 !
;  1092          1253  1 !       None.
;  1093          1254  1 !
;  1094          1255  1 ! Routine Value:
;  1095          1256  1 !
;  1096          1257  1 !       None.
;  1097          1258  1 !
;  1098          1259  1 ! Side Effects:
;  1099          1260  1 !
;  1100          1261  1 !       None.
;  1101          1262  1 !
;  1102          1263  1 !--
;  1103          1264  1
;  1104          1265  2     BEGIN
;  1105          1266  2
;  1106          1267  2     KEY_TABLE_(STR_OPT_OUTFMT);
;  1107          1268  2
;  1108          1269  2     LOCAL
;  1109          1270  2         NUMBER_ADR:     REF VECTOR[,WORD];
;  1110          1271  2
;  1111          1272  2     MACRO
;  1112        M 1273  2         GET_RFM_(KIND,VALU) =
;  1113        M 1274  2             BEGIN
;  1114        M 1275  2             %IF NOT %NULL(KIND)
;  1115        M 1276  2             %THEN
;  1116        M 1277  2                 RFM = %NAME('FAB$C_',KIND);
;  1117        M 1278  2             %FI
;  1118        M 1279  2             IF .SUBVAL_DESC[DSC$W_LENGTH] NEQ 0          ! If there is a value
;  1119        M 1280  2             THEN                                         ! Get optional number
;  1120        M 1281  2                 STORE_(VALU, .RSLT);                     ! Get Max Record Size
;  1121          1282  2             END %;
;  1122          1283  2
;  1123          1284  2
;  1124          1285  2     ! Determine which format was specified and set the appropriate
;  1125          1286  2     ! attributes in the output file area.
;  1126          1287  2     !
;  1127          1288  2     WHILE CLI_GET_VALUE(SD_FORM, STRING_DESC[BASE_]) DO
;  1128          1289  3         BEGIN
;  1129          1290  3         LOCAL
;  1130          1291  3             LOOKUP,
```

```
;  1131     1292  3              RSLT;
;  1132     1293
;  1133     1294                 ! Look up the keyword
;  1134     1295                 !
;  1135     1296                 LOOKUP = LOOKUP_KEY(KEYWORD_TABLE);
;  1136     1297
;  1137     1298                 ! If there is a value, get it.
;  1138     1299                 !
;  1139     1300                 IF .SUBVAL_DESC[DSC$W_LENGTH] NEQ 0
;  1140     1301                 THEN
;  1141     1302                     RSLT = NUMERIC_PARSE(SUBVAL_DESC[BASE_]);
;  1142     1303
;  1143     1304                 CASE .LOOKUP FROM 0 TO TABLE_SIZE-1 OF
;  1144     1305                     SET
;  1145     1306                     [K_FIXE]:   GET_RFM_(FIX,MRS);
;  1146     1307                     [K_VARI]:   GET_RFM_(VAR,MRS);
;  1147     1308                     [K_CONT]:   GET_RFM_(VFC,MRS);
;  1148     1309                     [K_SIZE]:   GET_RFM_(   ,FSZ);
;  1149     1310                     [K_BLOC]:   GET_RFM_(   ,BLS);
;  1150     1311                     [OUTRANGE]: 0;
;  1151     1312                     TES
;  1152     1313                 END;
;  1153     1314
;  1154     1315             ! Check for specifying SIZE and something other than CONTROLLED
;  1155     1316             !
;  1156     1317             IF .FSZ NEQ 0 AND .RFM NEQ 0 AND .RFM NEQ FAB$C_VFC
;  1157     1318             THEN
;  1158     1319                 CMD_ERROR(SOR$_KEYAMBINC);
;  1159     1320             2
;  1160     1321  1      END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                    002B4 P.ABV:    .ASCII  <5>\FIXED\                                  ;
                                                    002BA P.ABW:    .ASCII  <8>\VARIABLE\                               ;
                                                    002C3 P.ABX:    .ASCII  <10>\CONTROLLED\                            ;
                                                    002CE P.ABY:    .ASCII  <4>\SIZE\                                   ;
                                                    002D3 P.ABZ:    .ASCII  <10>\BLOCK_SIZE\                            ;
                                                    002DE           .BLKB   2                                           ;
                                                    002E0           .LONG   5                                           ;
00000000' 00000000' 00000000' 00000000' 00000000'   002E4 P.ABU:    .ADDRESS P.ABV, P.ABW, P.ABX, P.ABY, P.ABZ          ;

                                                                KEYWORD_TABLE=      P.ABU


                                                                .PSECT  $CODE$,NOWRT,2

                                                    000C 00000 SET_OUT_FMT:
                                                               .WORD    Save R2,R3                                      ; 1235
                                 53     0000'  CF   9E 00002           MOVAB    SUBVAL_DESC, R3                         ;
                                          F8   A3   9F 00007 1$:       PUSHAB   STRING_DESC                             ; 1288
                                       0000'  CF   9F 0000A            PUSHAB   SD_FORM                                 ;
                          0000G  CF                02   FB 0000E      CALLS    #2, CLI_GET_VALUE                       ;
                                 03                     50   E8 00013 BLBS     R0, 2$                                  ;
                                                   0082  31 00016     BRW      14$                                     ;
```

```
                                 0000'  CF  9F 00019 2$:    PUSHAB  KEYWORD_TABLE                   ; 1296
                          0000V  CF     01  FB 0001D         CALLS   #1, LOOKUP_KEY                  ;
                                 52     50  D0 00022         MOVL    R0, LOOKUP                      ;
                                        63  B5 00025         TSTW    SUBVAL_DESC                     ; 1300
                                        07  13 00027         BEQL    3$                              ;
                                        53  DD 00029         PUSHL   R3                              ; 1302
                          FF76  CF      01  FB 0002B         CALLS   #1, NUMERIC_PARSE               ;
                       04       00     52  CF 00030 3$:     CASEL   LOOKUP, #0, #4                  ; 1304
                0026        0018          0012 000C    00034 4$:     .WORD   6$-4$,-                  ;
                                                0046    0003C         7$-4$,-                         ;
                                                                      8$-4$,-                         ;
                                                                      10$-4$,-                        ;
                                                                      11$-4$                          ;
                                        C7  11 0003E 5$:     BRB     1$                              ;
                             09   A3        01  90 00040 6$:     MOVB    #1, RFM                     ; 1306
                                            0A  11 00044         BRB     9$                          ;
                             09   A3        02  90 00046 7$:     MOVB    #2, RFM                     ; 1307
                                            04  11 0004A         BRB     9$                          ;
                             09   A3        03  90 0004C 8$:     MOVB    #3, RFM                     ; 1308
                                        63  B5 00050 9$:     TSTW    SUBVAL_DESC                     ;
                                        B3  13 00052         BEQL    1$                              ;
                             0E   A3        50  B0 00054         MOVW    Z, MRS                      ;
                                            28  11 00058         BRB     12$                         ;
                                        63  B5 0005A 10$:    TSTW    SUBVAL_DESC                     ; 1309
                                        A9  13 0005C         BEQL    1$                              ;
                                 51     50  D0 0005E         MOVL    RSLT, Z                         ;
                             30   A3        51  90 00061         MOVB    Z, FSZ                      ;
                0051        0051     08     00  ED 00065         CMPZV   #0, #8, Z, Z                ;
                                            9B  13 0006A         BEQL    1$                          ;
                                            7E  D4 0006C         CLRL    -(SP)                       ;
                                            51  DD 0006E         PUSHL   Z                           ;
                                            02  DD 00070         PUSHL   #2                          ;
                                   001C1014 8F  DD 00072         PUSHL   #1839124                    ;
                                            1B  11 00078         BRB     13$                         ;
                                        63  B5 0007A 11$:    TSTW    SUBVAL_DESC                     ; 1310
                                        89  13 0007C         BEQL    1$                              ;
                             0C   A3        50  B0 0007E         MOVW    Z, BLS                      ;
                0050        0050     10     00  ED 00082 12$:    CMPZV   #0, #16, Z, Z               ;
                                            B5  13 00087         BEQL    5$                          ;
                                            7E  D4 00089         CLRL    -(SP)                       ;
                                            50  DD 0008B         PUSHL   Z                           ;
                                            02  DD 0008D         PUSHL   #2                          ;
                                   001C102C 8F  DD 0008F         PUSHL   #1839148                    ;
                          FAF5  CF      04  FB 00095 13$:    CALLS   #4, CMD_ERROR                   ;
                                            04  0009A         RET                                     ;
                                        30  A3  95 0009B 14$:    TSTB    FSZ                         ; 1317
                                            16  13 0009E         BEQL    15$                         ;
                                        09  A3  95 000A0         TSTB    RFM                         ;
                                            11  13 000A3         BEQL    15$                         ;
                                     03     09  A3  91 000A5         CMPB    RFM, #3                 ;
                                            0B  13 000A9         BEQL    15$                         ;
                                   001C8134 8F  DD 000AB         PUSHL   #1868084                    ; 1319
                          FAD9  CF      01  FB 000B1         CALLS   #1, CMD_ERROR                   ;
                                            04  000B6 15$:    RET                                     ; 1321

; Routine Size:  183 bytes,    Routine Base:  $CODE$ + 0471
```

```
;1162     1322  1 ROUTINE SET_PROC: NOVALUE =
;1163     1323  1
;1164     1324  1 !++
;1165     1325  1 ! Functional Description:
;1166     1326  1 !
;1167     1327  1 !       This routine sets the sort type to perform in the sort parameter area.
;1168     1328  1 !
;1169     1329  1 ! Formal Parameters:
;1170     1330  1 !
;1171     1331  1 !       The address of the qualifier descriptor block.
;1172     1332  1 !
;1173     1333  1 ! Implicit Inputs:
;1174     1334  1 !
;1175     1335  1 !       None.
;1176     1336  1 !
;1177     1337  1 ! Implicit Outputs:
;1178     1338  1 !
;1179     1339  1 !       None.
;1180     1340  1 !
;1181     1341  1 ! Routine Value:
;1182     1342  1 !
;1183     1343  1 !       None.
;1184     1344  1 !
;1185     1345  1 ! Side Effects:
;1186     1346  1 !
;1187     1347  1 !       None.
;1188     1348  1 !
;1189     1349  1 !--
;1190     1350  1
;1191     1351  2     BEGIN
;1192     1352  2
;1193     1353  2     KEY_TABLE_(STR_OPT_PROCESS);
;1194     1354  2
;1195     1355  2     OWN
;1196     1356  2         TAB: VECTOR[TABLE_SIZE,BYTE]
;1197     1357  2             PSECT($CODE$) PRESET(
;1198     1358  2                 [K_RECO]= SOR$GK_RECORD,
;1199     1359  2                 [K_TAG]=  SOR$GK_TAG,
;1200     1360  2                 [K_ADDR]= SOR$GK_ADDRESS,
;1201     1361  2                 [K_INDE]= SOR$GK_INDEX);
;1202     1362  2
;1203     1363  2     LOCAL
;1204     1364  2         LOOKUP;
;1205     1365  2
;1206     1366  2     ! Indicate that this qualifier was seen
;1207     1367  2     !
;1208     1368  2     FLAGS[IOS_PROC] = TRUE;
;1209     1369  2
;1210     1370  2     ! Determine which type was specified and set the appropriate attribute
;1211     1371  2     ! in the sort parameter area.
;1212     1372  2     !
;1213     1373  2     CLI_GET_VALUE(SD_PROC, STRING_DESC[BASE_]);
;1214     1374  2
;1215     1375  2     LOOKUP = LOOKUP_KEY(KEYWORD_TABLE);
;1216     1376  2     IF .LOOKUP LSSU TABLE_SIZE THEN TYP = .TAB[.LOOKUP];
;1217     1377  2
;1218     1378  2     IF .SUBVAL_DESC[DSC$W_LENGTH] NEQ 0
```

```
; 1219          1379  2     THEN                                    ! Value where none was expected
; 1220          1380  2         RETURN CMD_ERROR(SOR$_SHR_SYNTAX, 1, STRING_DESC[BASE_]);
; 1221          1381  2
; 1222          1382  1     END;


                                        00G 00* 00* 00* 00528 TAB:     .BYTE   <SOR$GK_RECORD&255>, <SOR$GK_TAG&255>, -      ;
                                                                       <SOR$GK_ADDRESS&255>, SOR$GK_INDEX                    ;

                                                                       .PSECT  $PLIT$,NOWRT,NOEXE,2

                                    44  52  4F  43  45  52  06  00528 P.ACB:   .ASCII  <6>\RECORD\                           ;
                                                47  41  47  03  002FF P.ACC:   .ASCII  <3>\TAG\                              ;
                                53  53  45  52  45  44  41  07  00303 P.ACD:   .ASCII  <7>\ADDRESS\                          ;
                                        58  45  44  4E  49  05  0030B P.ACE:   .ASCII  <5>\INDEX\                            ;
                                                                00311          .BLKB   3
                                                    00000004    00314          .LONG   4                                   ;
                    00000000' 00000000' 00000000' 00000000'     00318 P.ACA:   .ADDRESS P.ACB, P.ACC, P.ACD, P.ACE          ;

                                                                       KEYWORD_TABLE=      P.ACA


                                                                       .PSECT  $CODE$,NOWRT,2

                                                    0004 00000 SET_PROC:
                                                                       .WORD   Save R2                                     ; 1322
                                    52      0000'   CF  9E 00002       MOVAB   STRING_DESC, R2                             ;
                                 3E A2              01  88 00007       BISB2   #1, FLAGS+2                                 ; 1368
                                                    52  DD 0000B       PUSHL   R2                                          ; 1373
                                            0000'   CF  9F 0000D       PUSHAB  SD_PROC                                     ;
                                0000G   CF          02  FB 00011       CALLS   #2, CLI_GET_VALUE                           ;
                                            0000'   CF  9F 00016       PUSHAB  KEYWORD_TABLE                               ; 1375
                                0000V   CF          01  FB 0001A       CALLS   #1, LOOKUP_KEY                              ;
                                        04          50  D1 0001F       CMPL    LOOKUP, #4                                  ; 1376
                                  2D    A2          06  1E 00022       BGEQU   1$                                          ;
                                            D4 AF   40  90 00024       MOVB    TAB[LOOKUP], TYP                            ;
                                            08  A2  B5  0002A 1$:      TSTW    SUBVAL_DESC                                 ; 1378
                                                0F  13  0002D       BEQL    2$                                          ;
                                                52  DD  0002F       PUSHL   R2                                          ; 1380
                                                01  DD  00031       PUSHL   #1                                          ;
                                        001C10FC    8F  DD 00033       PUSHL   #1839356                                    ;
                                FA96    CF          03  FB 00039       CALLS   #3, CMD_ERROR                               ;
                                                    04  0003E 2$:      RET                                                 ; 1382

; Routine Size:  63 bytes,    Routine Base:  $CODE$ + 052C
```

```
;  1224          1383  1 ROUTINE SET_INP_FMT: NOVALUE =
;  1225          1384  1
;  1226          1385  1 !++
;  1227          1386  1 ! Functional Description:
;  1228          1387  1 !
;  1229          1388  1 !       This routine sets the input file record format in the input file
;  1230          1389  1 !       parameter area.
;  1231          1390  1 !
;  1232          1391  1 ! Formal Parameters:
;  1233          1392  1 !
;  1234          1393  1 !       The address of the qualifier descriptor block.
;  1235          1394  1 !
;  1236          1395  1 ! Implicit Inputs:
;  1237          1396  1 !
;  1238          1397  1 !       None.
;  1239          1398  1 !
;  1240          1399  1 ! Implicit Outputs:
;  1241          1400  1 !
;  1242          1401  1 !       None.
;  1243          1402  1 !
;  1244          1403  1 ! Routine Value:
;  1245          1404  1 !
;  1246          1405  1 !       None.
;  1247          1406  1 !
;  1248          1407  1 ! Side Effects:
;  1249          1408  1 !
;  1250          1409  1 !       None.
;  1251          1410  1 !
;  1252          1411  1 !--
;  1253          1412  1
;  1254          1413  2     BEGIN
;  1255          1414  2
;  1256          1415  2     KEY_TABLE_(STR_OPT_INPFMT);
;  1257          1416  2
;  1258          1417  2
;  1259          1418  2     ! Get values and store them away.
;  1260          1419  2     !
;  1261          1420  2     WHILE CLI_GET_VALUE(SD_FORM, STRING_DESC[BASE_]) DO
;  1262          1421  3         BEGIN
;  1263          1422  3
;  1264          1423  3         CASE LOOKUP_KEY(KEYWORD_TABLE) FROM 0 TO TABLE_SIZE-1 OF
;  1265          1424  3             SET
;  1266          1425  3             [K_RECO]:
;  1267          1426  3                 STORE_(LRL, NUMERIC_PARSE(SUBVAL_DESC[BASE_]));
;  1268          1427  3             [K_FILE]:
;  1269          1428  3                 STORE_(INP_ALQ, .INP_ALQ + NUMERIC_PARSE(SUBVAL_DESC[BASE_]));
;  1270          1429  3             [OUTRANGE]:
;  1271          1430  3                 0;
;  1272          1431  3             TES;
;  1273          1432  3
;  1274          1433  2         END;
;  1275          1434  2
;  1276          1435  1     END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2
```

```
                45  5A  49  53  5F  45  4C  49  46  09  00328 P.ACG:  .ASCII  <9>\FILE_SIZE\                    ;
            45  5A  49  53  5F  44  52  4F  43  45  52  0B  00332 P.ACH:  .ASCII  <11>\RECORD_SIZE\              ;
                                                        0033E          .BLKB   2                                 ;
                                                00000002  00340          .LONG   2                                 ;
                                    00000000'   00000000' 00344 P.ACF:  .ADDRESS P.ACG, P.ACH                     ;

                                                                        KEYWORD_TABLE=      P.ACF


                                                                        .PSECT  $CODE$,NOWRT,2

                                                        0004 00000 SET_INP_FMT:
                                                                        .WORD   Save R2                           ; 1383
                                        52          0000'   CF   9E 00002          MOVAB   SUBVAL_DESC, R2       ;
                                                      F8    A2   9F 00007 1$:       PUSHAB  STRING_DESC           ; 1420
                                                    0000'   CF   9F 0000A          PUSHAB  SD_FORM               ;
                                    0000G   CF              02   FB 0000E          CALLS   #2, CLI_GET_VALUE     ;
                                            44              50   E9 00013          BLBC    R0, 5$                ;
                                                    0000'   CF   9F 00016          PUSHAB  KEYWORD_TABLE         ; 1423
                                    0000V   CF              01   FB 0001A          CALLS   #1, LOOKUP_KEY        ;
                        01                  00              50   CF 0001F          CASEL   R0, #0, #1            ;
                                          0006                002A     00023 2$:    .WORD   4$-2$,-               ;
                                                                                               3$-2$              ;
                                                              DE   11 00027          BRB     1$                    ;
                                                              52   DD 00029 3$:     PUSHL   R2                    ; 1426
                                    FE7C    CF                01   FB 0002B          CALLS   #1, NUMERIC_PARSE     ;
                                      1C    A2                50   B0 00030          MOVW    Z, LRL                ;
                50                  50                      00   ED 00034          CMPZV   #0, #16, Z, Z         ;
                                            10                CC   13 00039          BEQL    1$                    ;
                                                              7E   D4 0003B          CLRL    -(SP)                 ;
                                                              50   DD 0003D          PUSHL   Z                     ;
                                                              02   DD 0003F          PUSHL   #2                    ;
                                                  001C102C    8F   DD 00041          PUSHL   #1839148              ;
                                    FA49    CF                04   FB 00047          CALLS   #4, CMD_ERROR         ;
                                                              04   04 0004C          RET                           ;
                                                              52   DD 0004D 4$:     PUSHL   R2                    ; 1428
                                    FE58    CF                01   FB 0004F          CALLS   #1, NUMERIC_PARSE     ;
                                      20    A2                50   C0 00054          ADDL2   R0, INP_ALQ           ;
                                                              AD   11 00058          BRB     1$                    ; 1429
                                                                   04 0005A 5$:      RET                           ; 1433

; Routine Size:  91 bytes,    Routine Base:  $CODE$ + 056B
```

```
;  1278          1436  1 ROUTINE NOT_WORK: NOVALUE =
;  1279          1437  1
;  1280          1438  1 !++
;  1281          1439  1 ! Functional Description:
;  1282          1440  1 !
;  1283          1441  1 !       This routine is called if the user specifies /NOWORK_FILES
;  1284          1442  1 !
;  1285          1443  1 ! Formal Parameters:
;  1286          1444  1 !
;  1287          1445  1 !       The address of the qualifier descriptor block.
;  1288          1446  1 !
;  1289          1447  1 ! Implicit Inputs:
;  1290          1448  1 !
;  1291          1449  1 !       None.
;  1292          1450  1 !
;  1293          1451  1 ! Implicit Outputs:
;  1294          1452  1 !
;  1295          1453  1 !       None.
;  1296          1454  1 !
;  1297          1455  1 ! Routine Value:
;  1298          1456  1 !
;  1299          1457  1 !       None.
;  1300          1458  1 !
;  1301          1459  1 ! Side Effects:
;  1302          1460  1 !
;  1303          1461  1 !       None.
;  1304          1462  1 !
;  1305          1463  1 !--
;  1306          1464  1
;  1307          1465  2     BEGIN
;  1308          1466  2
;  1309          1467  2     ! Make sure that the user explicitly requested no work files.
;  1310          1468  2     !
;  1311          1469  2     IF CLI_PRESENT( SD_WORK ) EQL CLI$_NEGATED
;  1312          1470  2     THEN
;  1313          1471  3         BEGIN
;  1314          1472  3         !
;  1315          1473  3         ! Act as though the user specified 0 work files.
;  1316          1474  3         !
;  1317          1475  3         FLAGS[IOS_WORK] = TRUE;
;  1318          1476  3         WRK = 0;
;  1319          1477  2         END;
;  1320          1478  1     END;


                                        0000 00000 NOT_WORK:
                                                           .WORD   Save nothing                    ; 1436
                                0000'   CF  9F 00002       PUSHAB  SD_WORK                         ; 1469
    0000G   CF                          01  FB 00006       CALLS   #1, CLI_PRESENT                 ;
00000000G   8F                          50  D1 0000B       CMPL    R0, #CLI$_NEGATED               ;
                                        09  12 00012       BNEQ    1$                              ;
    0000'   CF                          01  88 00014       BISB2   #1, FLAGS+1                     ; 1475
                                0000'   CF  94 00019       CLRB    WRK                             ; 1476
                                            04 0001D 1$:   RET                                     ; 1478
```

; Routine Size: 30 bytes,   Routine Base: $CODE$ + 05C6

```
;  1322          1479  1 ROUTINE SET_WORK: NOVALUE =
;  1323          1480  1
;  1324          1481  1 !++
;  1325          1482  1 ! Functional Description:
;  1326          1483  1 !
;  1327          1484  1 !       This routine gets the number of work files to use.
;  1328          1485  1 !
;  1329          1486  1 ! Formal Parameters:
;  1330          1487  1 !
;  1331          1488  1 !       The address of the qualifier descriptor block.
;  1332          1489  1 !
;  1333          1490  1 ! Implicit Inputs:
;  1334          1491  1 !
;  1335          1492  1 !       None.
;  1336          1493  1 !
;  1337          1494  1 ! Implicit Outputs:
;  1338          1495  1 !
;  1339          1496  1 !       None.
;  1340          1497  1 !
;  1341          1498  1 ! Routine Value:
;  1342          1499  1 !
;  1343          1500  1 !       None.
;  1344          1501  1 !
;  1345          1502  1 ! Side Effects:
;  1346          1503  1 !
;  1347          1504  1 !       None.
;  1348          1505  1 !
;  1349          1506  1 !--
;  1350          1507  1
;  1351          1508  2     BEGIN
;  1352          1509  2
;  1353          1510  2     ! If there is a value, store it into WRK.
;  1354          1511  2     ! Otherwise, let the sort routines default this value.
;  1355          1512  2     !
;  1356          1513  2     IF CLI_GET_VALUE( SD_WORK, STRING_DESC[BASE_] )
;  1357          1514  2     THEN
;  1358          1515  3         STORE_(WRK, NUMERIC_PARSE(STRING_DESC[BASE_]))
;  1359          1516  2     ELSE
;  1360          1517  2         FLAGS[IOS_WORK] = FALSE;
;  1361          1518  2
;  1362          1519  1     END;


                                        0004 00000 SET_WORK:
                                                           .WORD   Save R2                                 ; 1479
                          52    0000'   CF  9E 00002            MOVAB   STRING_DESC, R2                 ;
                                        52  DD 00007            PUSHL   R2                              ; 1513
                                0000'   CF  9F 00009            PUSHAB  SD_WORK                         ;
                    0000G       CF      02  FB 0000D            CALLS   #2, CLI_GET_VALUE               ;
                                24      50  E9 00012            BLBC    R0, 1$                          ;
                                        52  DD 00015            PUSHL   R2                              ; 1515
                    FE17        CF      01  FB 00017            CALLS   #1, NUMERIC_PARSE               ;
                      2C        A2      50  90 0001C            MOVB    Z, WRK                          ;
          50            50      08      00  ED 00020            CMPZV   #0, #8, Z, Z                    ;
```

```
                                     16  13 00025          BEQL    2$                                      ;
                                     7E  D4 00027          CLRL    -(SP)                                   ;
                                     50  DD 00029          PUSHL   Z                                       ;
                                     02  DD 0002B          PUSHL   #2                                      ;
                          001C1014   8F  DD 0002D          PUSHL   #1839124                                ;
                F9E4   CF            04  FB 00033          CALLS   #4, CMD_ERROR                           ;
                                         04 00038          RET                                             ;
                  3D   A2            01  8A 00039 1$:      BICB2   #1, FLAGS+1                             ; 1517
                                         04 0003D 2$:      RET                                             ; 1519

; Routine Size:  62 bytes,    Routine Base:  $CODE$ + 05E4
```

```
;  1364          1520  1 ROUTINE SET_BUCK: NOVALUE =
;  1365          1521  1
;  1366          1522  1 !++
;  1367          1523  1 ! Functional Description:
;  1368          1524  1 !
;  1369          1525  1 !       This routine gets the bucket size to use.
;  1370          1526  1 !
;  1371          1527  1 ! Formal Parameters:
;  1372          1528  1 !
;  1373          1529  1 !       The address of the qualifier descriptor block.
;  1374          1530  1 !
;  1375          1531  1 ! Implicit Inputs:
;  1376          1532  1 !
;  1377          1533  1 !       None.
;  1378          1534  1 !
;  1379          1535  1 ! Implicit Outputs:
;  1380          1536  1 !
;  1381          1537  1 !       None.
;  1382          1538  1 !
;  1383          1539  1 ! Routine Value:
;  1384          1540  1 !
;  1385          1541  1 !       None.
;  1386          1542  1 !
;  1387          1543  1 ! Side Effects:
;  1388          1544  1 !
;  1389          1545  1 !       None.
;  1390          1546  1 !
;  1391          1547  1 !--
;  1392          1548  1
;  1393          1549  2     BEGIN
;  1394          1550  2     CLI_GET_VALUE( SD_BUCK, STRING_DESC[BASE_] );
;  1395          1551  2     STORE_(BKS, NUMERIC_PARSE(STRING_DESC[BASE_]));
;  1396          1552  1     END;


                                        0000 00000 SET_BUCK:
                                                           .WORD   Save nothing                    ;  1520
                                0000'   CF   9F 00002      PUSHAB  STRING_DESC                     ;  1550
                                0000'   CF   9F 00006      PUSHAB  SD_BUCK                         ;
                        0000G   CF           02 FB 0000A   CALLS   #2, CLI_GET_VALUE               ;
                                0000'   CF   9F 0000F      PUSHAB  STRING_DESC                     ;  1551
                        FDDD    CF           01 FB 00013   CALLS   #1, NUMERIC_PARSE               ;
                        0000'   CF           50 90 00018   MOVB    Z, BKS                          ;
        50              50      08           00 ED 0001D   CMPZV   #0, #8, Z, Z                    ;
                                             11 13 00022   BEQL    1$                              ;
                                             7E D4 00024   CLRL    -(SP)                           ;
                                             50 DD 00026   PUSHL   Z                               ;
                                             02 DD 00028   PUSHL   #2                              ;
                                001C1014     8F DD 0002A   PUSHL   #1839124                        ;
                        F9A9    CF           04 FB 00030   CALLS   #4, CMD_ERROR                   ;
                                             04 00035 1$:  RET                                     ;  1552

; Routine Size:  54 bytes,    Routine Base:  $CODE$ + 0622
```

```
;  1398          1553  1 ROUTINE SET_ALLO: NOVALUE =
;  1399          1554  1
;  1400          1555  1 !++
;  1401          1556  1 ! Functional Description:
;  1402          1557  1 !
;  1403          1558  1 !       This routine gets the output file allocation to use.
;  1404          1559  1 !
;  1405          1560  1 ! Formal Parameters:
;  1406          1561  1 !
;  1407          1562  1 !       The address of the qualifier descriptor block.
;  1408          1563  1 !
;  1409          1564  1 ! Implicit Inputs:
;  1410          1565  1 !
;  1411          1566  1 !       None.
;  1412          1567  1 !                                                                                    ; F
;  1413          1568  1 ! Implicit Outputs:
;  1414          1569  1 !
;  1415          1570  1 !       None.
;  1416          1571  1 !
;  1417          1572  1 ! Routine Value:
;  1418          1573  1 !
;  1419          1574  1 !       None.
;  1420          1575  1 !
;  1421          1576  1 ! Side Effects:
;  1422          1577  1 !
;  1423          1578  1 !       None.
;  1424          1579  1 !
;  1425          1580  1 !--
;  1426          1581  1
;  1427          1582  2     BEGIN
;  1428          1583  2     CLI_GET_VALUE( SD_ALLO, STRING_DESC[BASE_] );
;  1429          1584  2     STORE_(OUT_ALQ, NUMERIC_PARSE(STRING_DESC[BASE_]));
;  1430          1585  1     END;


                                        0000 00000 SET_ALLO:
                                                           .WORD   Save nothing                          ; 1553
                                0000'   CF   9F 00002      PUSHAB  STRING_DESC                           ; 1583
                                0000'   CF   9F 00006      PUSHAB  SD_ALLO                               ;
                          0000G CF           02 FB 0000A   CALLS   #2, CLI_GET_VALUE                     ;
                                0000'   C    9F 0000F      PUSHAB  STRING_DESC                           ; 1584
                          FDA7  CF           01 FB 00013   CALLS   #1, NUMERIC_PARSE                     ;
                          0000' CF           50 D0 00018   MOVL    R0, OUT_ALQ                           ;
                                             04 0001D      RET                                           ; 1585

; Routine Size:  30 bytes,    Routine Base:  $CODE$ + 0658
```

```
;  1432          1586  1 ROUTINE PARSE_KEY: NOVALUE =
;  1433          1587  1
;  1434          1588  1 !++
;  1435          1589  1 ! Functional Description:
;  1436          1590  1 !
;  1437          1591  1 !       This routine parses the key definition and stores it in the key
;  1438          1592  1 !       buffer.
;  1439          1593  1 !
;  1440          1594  1 ! Formal Parameters:
;  1441          1595  1 !
;  1442          1596  1 !       The address of the qualifier descriptor block.
;  1443          1597  1 !
;  1444          1598  1 ! Implicit Inputs:
;  1445          1599  1 !
;  1446          1600  1 !       The key buffer.
;  1447          1601  1 !
;  1448          1602  1 ! Implicit Outputs:
;  1449          1603  1 !
;  1450          1604  1 !       None.
;  1451          1605  1 !
;  1452          1606  1 ! Routine Value:
;  1453          1607  1 !
;  1454          1608  1 !       None.
;  1455          1609  1 !
;  1456          1610  1 ! Side Effects:
;  1457          1611  1 !
;  1458          1612  1 !       None.
;  1459          1613  1 !
;  1460          1614  1 !--
;  1461          1615  1
;  1462          1616  2     BEGIN
;  1463          1617  2
;  1464          1618  2     OWN
;  1465          1619  2         KEY_NUM:        BYTE INITIAL (0);
;  1466          1620  2
;  1467          1621  2     BUILTIN
;  1468          1622  2         TESTBITSC, TESTBITCC,
;  1469          1623  2         TESTBITSS, TESTBITCS;
;  1470          1624  2
;  1471          1625  2     LOCAL
;  1472          1626  2         CURRENT_KEY:    REF BLOCK;
;  1473          1627  2
;  1474          1628  2     LITERAL                 ! Bits for OPTS
;  1475          1629  2         B_SEP    = 0,       ! Separate
;  1476          1630  2         B_OVE    = 1,       ! Overpunched
;  1477          1631  2         B_LEA    = 2,       ! Leading
;  1478          1632  2         B_TRA    = 3,       ! Trailing
;  1479          1633  2         B_UNS    = 4,       ! Unsigned
;  1480          1634  2         B_SIG    = 5,       ! Signed
;  1481          1635  2         B_ASC    = 6,       ! Ascending
;  1482          1636  2         B_DES    = 7,       ! Descending
;  1483          1637  2         B_POS    = 8,       ! Position
;  1484          1638  2         B_SIZ    = 9,       ! Size
;  1485          1639  2         B_NUM    = 10       ! Number
;  1486        U 1640  2 %IF FUN_K_KANJI %THEN ,
;  1487        U 1641  2         B_DTY    = 11       ! Dtype
;  1488          1642  2 %FI      ;
```

```
;  1489          1643  2
;  1490          1644  2      KEY_TABLE_(STR_OPT_KEY);
;  1491          1645  2
;  1492          1646  2      OWN
;  1493          1647  2          TAB: VECTOR[TABLE_SIZE,BYTE]
;  1494          1648  2              PSECT($CODE$) PRESET(
;  1495          1649  2                  [K_ASCE]=       B_ASC,
;  1496          1650  2                  [K_DESC]=       B_DES,
;  1497          1651  2                  [K_BINA]=       DSC$K_DTYPE_B,          ! We fix this later
;  1498          1652  2                  [K_CHAR]=       DSC$K_DTYPE_T,
;  1499          1653  2                  [K_DECI]=       DSC$K_DTYPE_NRO,        ! We fix this later
;  1500          1654  2                  [K_PACK]=       DSC$K_DTYPE_P,
;  1501          1655  2                  [K_ZONE]=       DSC$K_DTYPE_NZ,
;  1502          1656  2                  [K_F_FL]=       DSC$K_DTYPE_F,
;  1503          1657  2                  [K_D_FL]=       DSC$K_DTYPE_D,
;  1504          1658  2                  [K_G_FL]=       DSC$K_DTYPE_G,
;  1505          1659  2                  [K_H_FL]=       DSC$K_DTYPE_H,
;  1506          1660  2                  [K_SIGN]=       B_SIG,
;  1507          1661  2                  [K_UNSI]=       B_UNS,
;  1508          1662  2                  [K_LEAD]=       B_LEA,
;  1509          1663  2                  [K_TRAI]=       B_TRA,
;  1510          1664  2                  [K_OVER]=       B_OVE,
;  1511          1665  2                  [K_SEPA]=       B_SEP);
;  1512          1666  2      LOCAL                                   ! Temporary storage for key
;  1513          1667  2          TYPE:   %IF FUN_K_KANJI %THEN WORD %ELSE BYTE %FI,
;  1514          1668  2          POSIT:  WORD,
;  1515          1669  2          SIZ:    WORD,
;  1516          1670  2          ASCDES: BYTE,
;  1517          1671  2          OPTS:   BITVECTOR[32];          ! Key parser flags
;  1518          1672  2
;  1519          1673  2      KEY_NUM     = .KEY_NUM + 1;         ! Set up defaults
;  1520          1674  2      TYPE        = 0;
;  1521          1675  2      POSIT       = 1;                    ! Default position
;  1522          1676  2      SIZ         = -1;                   ! Default size (entire record)
;  1523          1677  2      OPTS        = 0;
;  1524          1678  2
;  1525          1679  2      WHILE CLI_GET_VALUE(SD_KEY, STRING_DESC[BASE_]) DO
;  1526          1680  3          BEGIN
;  1527          1681  3          LOCAL
;  1528          1682  3              LOOKUP;
;  1529          1683  3
;  1530          1684  3          LOOKUP = LOOKUP_KEY(KEYWORD_TABLE);
;  1531          1685  3          IF .LOOKUP EQL K_SI
;  1532          1686  3          THEN
;  1533          1687  4              BEGIN
;  1534          1688  4              CMD_ERROR(SOR$_SHR_BADKEY_W, 1, STRING_DESC[BASE_]);
;  1535          1689  4              IF .SUBVAL_DESC[DSC$W_LENGTH] NEQ 0
;  1536          1690  4              THEN LOOKUP = K_SIZE
;  1537          1691  4              ELSE LOOKUP = K_SIGN;
;  1538          1692  3              END;
;  1539          1693  3
;  1540        P 1694  3          IF NOT ONEOF (.LOOKUP, BMSK_(K_POSI, K_SIZE, K_NUMB
;  1541          1695  4              %IF FUN_K_KANJI %THEN , K_DTYP %FI)
;  1542          1696  3          THEN
;  1543          1697  4              BEGIN
;  1544          1698  4              IF .SUBVAL_DESC[DSC$W_LENGTH] NEQ 0
;  1545          1699  4              THEN                                ! Value where none was expected
```

```
;  1546          1700  4                RETURN CMD_ERROR(SOR$_SHR_SYNTAX, 1, STRING_DESC[BASE_]);
;  1547          1701  3            END;
;  1548          1702  3
;  1549          1703  3
;  1550          1704  3        ! Process the keyword.
;  1551          1705  3        !
;  1552          1706  3        IF
;  1553        P 1707  3            ONEOF_(.LOOKUP, BMSK_(K_BINA,K_CHAR,K_DECI,K_PACK,K_ZONE,
;  1554          1708  4                K_F_FL,K_D_FL,K_G_FL,K_H_FL))
;  1555          1709  3        THEN
;  1556          1710  4            BEGIN
;  1557          1711  4            !
;  1558          1712  4            ! Check whether the user has already specified the key data type.
;  1559          1713  4            !
;  1560          1714  4            IF .TYPE NEQ 0 THEN CMD_ERROR(SOR$_KEYAMBINC);
;  1561          1715  4            !
;  1562          1716  4            ! Set the key data type.  BINARY and DECIMAL are fixed up later.
;  1563          1717  4            !
;  1564          1718  4            TYPE = .TAB[.LOOKUP];
;  1565          1719  4            END
;  1566          1720  3        ELIF
;  1567        P 1721  3            ONEOF_(.LOOKUP, BMSK_(K_ASCE,K_DESC,K_SIGN,K_UNSI,K_LEAD,K_TRAI,
;  1568          1722  4                K_OVER,K_SEPA))
;  1569          1723  3        THEN
;  1570          1724  4            BEGIN
;  1571          1725  4            !
;  1572          1726  4            ! Set the option, and verify that it's opposite isn't set.
;  1573          1727  4            !
;  1574          1728  4            ASSERT_((B_ASC XOR B_DES) EQL 1)
;  1575          1729  4            ASSERT_((B_SIG XOR B_UNS) EQL 1)
;  1576          1730  4            ASSERT_((B_LEA XOR B_TRA) EQL 1)
;  1577          1731  4            ASSERT_((B_OVE XOR B_SEP) EQL 1)
;  1578          1732  4            OPTS[.TAB[.LOOKUP]] = TRUE;
;  1579          1733  4            IF TESTBITSC(OPTS[.TAB[.LOOKUP] XOR 1])
;  1580          1734  4            THEN
;  1581          1735  4                CMD_ERROR(SOR$_KEYAMBINC);
;  1582          1736  4            END
;  1583          1737  3        ELIF
;  1584          1738  3            .LOOKUP EQL K_POSI
;  1585          1739  3        THEN
;  1586          1740  4            BEGIN                                   ! Set starting position of key
;  1587          1741  4            IF TESTBITSS(OPTS[B_POS]) THEN CMD_ERROR(SOR$_KEYAMBINC);
;  1588          1742  4            STORE_(POSIT, NUMERIC_PARSE(SUBVAL_DESC[BASE_]));
;  1589          1743  4            END
;  1590          1744  3        ELIF
;  1591          1745  3            .LOOKUP EQL K_SIZE
;  1592          1746  3        THEN
;  1593          1747  4            BEGIN                                   ! Set size of key field
;  1594          1748  4            LOCAL
;  1595          1749  4                TMP;
;  1596          1750  4            IF TESTBITSS(OPTS[B_SIZ]) THEN CMD_ERROR(SOR$_KEYAMBINC);
;  1597          1751  4            TMP = NUMERIC_PARSE(SUBVAL_DESC[BASE_]);
;  1598          1752  4            IF .TMP EQL -1 THEN SIZ = -1 ELSE STORE_(SIZ, .TMP);
;  1599          1753  4            END
;  1600          1754  3        ELIF
;  1601          1755  3            .LOOKUP EQL K_NUMB
;  1602          1756  3        THEN
```

```
;1603          1757  4              BEGIN                                   ! Set key number
;1604          1758  4              IF TESTBITSS(OPTS[B_NUM]) THEN CMD_ERROR(SOR$_KEYAMBINC);
;1605          1759  4              STORE_(KEY_NUM, NUMERIC_PARSE(SUBVAL_DESC[BASE_]));
;1606          1760  4              END
;1607        U 1761  4 %IF FUN_K_KANJI %THEN
;1608        U 1762  4          EIF
;1609        U 1763  4              .LOOKUP EQL K_DTYP
;1610        U 1764  4          THEN
;1611        U 1765  4              BEGIN
;1612        U 1766  4              !
;1613        U 1767  4              ! Convert the string to a number
;1614        U 1768  4              !
;1615        U 1769  4              LOCAL
;1616        U 1770  4                  STATUS;
;1617        U 1771  4              IF SOR$$DTYPE_T_W EQL 0
;1618        U 1772  4              THEN
;1619        U 1773  4                  CMD_ERROR(SOR$_SHR_BADKEY, 1, STRING_DESC[BASE_]);
;1620        U 1774  4              IF .TYPE NEQ 0 THEN CMD_ERROR(SOR$_KEYAMBINC);
;1621        U 1775  4              IF .SUBVAL_DESC[DSC$W_LENGTH] EQL 0
;1622        U 1776  4              THEN                                    ! No value where one was expected
;1623        U 1777  4                  CMD_ERROR(SOR$_SHR_NOVALUE, 1, STRING_DESC[BASE_]);
;1624        U 1778  4
;1625        U 1779  4              STATUS = SOR$$DTYPE_T_W(SUBVAL_DESC[BASE_], TYPE);
;1626        U 1780  4              IF NOT .STATUS THEN CMD_ERROR(SOR$_KEYAMBINC, 0, .STATUS);
;1627        U 1781  4              OPTS[B_DTY] = TRUE;
;1628        U 1782  4              END
;1629          1783  4 %FI
;1630          1784  3          ELSE
;1631          1785  4              BEGIN
;1632          1786  4              !
;1633          1787  4              ! We've already complained about other keywords.
;1634          1788  4              ! K_SI has already been converted to K_SIZE or K_SIGN.
;1635          1789  4              !
;1636          1790  4              0;
;1637          1791  3              END;
;1638          1792  2          END;
;1639          1793  2
;1640          1794  2
;1641        U 1795  2 %IF FUN_K_KANJI %THEN
;1642        U 1796  2      IF NOT .OPTS[B_DTY] THEN
;1643          1797  2 %FI
;1644          1798  2      IF .TYPE EQL 0 THEN TYPE = DSC$K_DTYPE_T;   ! Default to text
;1645          1799  2
;1646          1800  2
;1647          1801  2      ! Compute the size for the floating datatypes
;1648          1802  2      !
;1649        P 1803  2      IF ONEOF (.TYPE, BMSK_(
;1650          1804  3          DSC$K_DTYPE_F, DSC$K_DTYPE_D, DSC$K_DTYPE_G, DSC$K_DTYPE_H))
;1651          1805  2      THEN
;1652          1806  3          BEGIN
;1653          1807  3          LOCAL
;1654          1808  3              TMP;        ! Save the user-specified size for a moment
;1655          1809  3          ASSERT_(1^(DSC$K_DTYPE_F MOD 8) EQL 4)
;1656          1810  3          ASSERT_(1^(DSC$K_DTYPE_D MOD 8) EQL 8)
;1657          1811  3          ASSERT_(1^(DSC$K_DTYPE_G MOD 8) EQL 8)
;1658          1812  3          ASSERT_(1^(DSC$K_DTYPE_H MOD 8) EQL 16)
;1659          1813  3          TMP = .SIZ;
```

```
;  1660          1814  3          SIZ = 1 ^ .TYPE<0,3,0>;
;  1661          1815  3
;  1662          1816  3          ! If the user specified a size, make sure he was correct
;  1663          1817  3          !
;  1664          1818  3          IF TESTBITSS(OPTS[B_SIZ])
;  1665          1819  3          THEN
;  1666          1820  3              IF .TMP NEQ .SIZ THEN CMD_ERROR(SOR$_BAD_KEY);
;  1667          1821  2          END;
;  1668          1822  2
;  1669        U 1823  2 %IF FUN_K_KANJI %THEN
;  1670        U 1824  2      IF .OPTS[B_DTY]
;  1671        U 1825  2      THEN
;  1672        U 1826  2          BEGIN
;  1673        U 1827  2          OPTS[B_POS] = TRUE;      ! Assume position seen
;  1674        U 1828  2          OPTS[B_SIZ] = TRUE;      ! Assume size seen
;  1675        U 1829  2          END;
;  1676          1830  2 %FI
;  1677          1831  2
;  1678          1832  2      ! Make sure that position and size were specified
;  1679          1833  2      !
;  1680          1834  2      IF TESTBITSC(OPTS[B_POS]) AND TESTBITSC(OPTS[B_SIZ])
;  1681          1835  2      THEN
;  1682          1836  2          0
;  1683          1837  2      ELSE
;  1684          1838  2          CMD_ERROR(SOR$_BAD_KEY);
;  1685          1839  2
;  1686          1840  2
;  1687          1841  2      ! Check the number of keys
;  1688          1842  2      !
;  1689          1843  2      IF .KEY_NUM EQL 0 OR .KEY_NUM GTRU MAX_KEYS
;  1690          1844  2      THEN
;  1691          1845  3          BEGIN
;  1692          1846  3          CMD_ERROR(SOR$_NUM_KEY);
;  1693          1847  3          KEY_NUM = MAX_KEYS;
;  1694          1848  2          END;
;  1695          1849  2      OPTS[B_NUM] = FALSE;
;  1696          1850  2
;  1697          1851  2
;  1698          1852  2      ! Some checks on the data type
;  1699          1853  2      !
;  1700        P 1854  2      IF  ONEOF_(.TYPE, BMSK_(DSC$K_DTYPE_T, DSC$K_DTYPE_NZ, DSC$K_DTYPE_P,
;  1701          1855  3                  DSC$K_DTYPE_F, DSC$K_DTYPE_D, DSC$K_DTYPE_G, DSC$K_DTYPE_H))
;  1702          1856  2      THEN
;  1703          1857  2          0
;  1704          1858  2      ELIF
;  1705          1859  2          .TYPE EQL DSC$K_DTYPE_B
;  1706          1860  2      THEN
;  1707          1861  3          BEGIN
;  1708          1862  3          IF    .SIZ EQL  1 THEN 0
;  1709          1863  3          ELIF  .SIZ EQL  2 THEN TYPE = DSC$K_DTYPE_W
;  1710          1864  3          ELIF  .SIZ EQL  4 THEN TYPE = DSC$K_DTYPE_L
;  1711          1865  3          ELIF  .SIZ EQL  8 THEN TYPE = DSC$K_DTYPE_Q
;  1712          1866  3          ELIF  .SIZ EQL 16 THEN TYPE = DSC$K_DTYPE_O
;  1713          1867  3          ELSE CMD_ERROR(SOR$_BAD_KEY);
;  1714          1868  3          IF .OPTS[B_UNS]
;  1715          1869  3          THEN
;  1716          1870  4              BEGIN
```

```
;  1717          1871  4              ASSERT_(DSC$K_DTYPE_B-4 EQL DSC$K_DTYPE_BU)
;  1718          1872  4              ASSERT_(DSC$K_DTYPE_W-4 EQL DSC$K_DTYPE_WU)
;  1719          1873  4              ASSERT_(DSC$K_DTYPE_L-4 EQL DSC$K_DTYPE_LU)
;  1720          1874  4              ASSERT_(DSC$K_DTYPE_Q-4 EQL DSC$K_DTYPE_QU)
;  1721          1875  4              IF .TYPE EQL DSC$K_DTYPE_O
;  1722          1876  4                  THEN TYPE = DSC$K_DTYPE_OU
;  1723          1877  4                  ELSE TYPE = .TYPE - 4;  ! Convert from DTYPE_x to DTYPE_xU
;  1724          1878  3              END;
;  1725          1879  3          OPTS[B_SIG] = FALSE;
;  1726          1880  3          OPTS[B_UNS] = FALSE;
;  1727          1881  3          END
;  1728          1882  2      ELIF
;  1729          1883  2          .TYPE EQL DSC$K_DTYPE_NRO
;  1730          1884  2      THEN
;  1731          1885  3          BEGIN
;  1732          1886  3          IF TESTBITSC(OPTS[B_UNS])
;  1733          1887  3          THEN
;  1734          1888  3              TYPE = DSC$K_DTYPE_NU
;  1735          1889  3          ELSE
;  1736          1890  4              BEGIN
;  1737          1891  4              OPTS[B_SIG] = FALSE;
;  1738          1892  4              IF .OPTS[B_LEA] THEN TYPE = DSC$K_DTYPE_NLO;
;  1739          1893  4              OPTS[B_LEA] = FALSE;
;  1740          1894  4              OPTS[B_TRA] = FALSE;
;  1741          1895  4              IF .OPTS[B_SEP]
;  1742          1896  4              THEN
;  1743          1897  5                  BEGIN
;  1744          1898  5                  ASSERT_(DSC$K_DTYPE_NLO - 1 EQL DSC$K_DTYPE_NL)
;  1745          1899  5                  ASSERT_(DSC$K_DTYPE_NRO - 1 EQL DSC$K_DTYPE_NR)
;  1746          1900  5                  TYPE = .TYPE - 1;       ! Convert from DTYPE_NxO to DTYPE_Nx
;  1747          1901  5                  SIZ = .SIZ + 1;         ! Increase the size to include the sign
;  1748          1902  4                  END;
;  1749          1903  4              OPTS[B_SEP] = FALSE;
;  1750          1904  4              OPTS[B_OVE] = FALSE;
;  1751          1905  4              END
;  1752          1906  3          END
;  1753          1907  3      %IF FUN_K_KANJI %THEN ELSE IF TESTBITSC(OPTS[B_DTY]) THEN 0 %FI
;  1754          1908  2      ELSE
;  1755          1909  2          CMD_ERROR(SOR$_SHR_BADLOGIC);
;  1756          1910  2
;  1757          1911  2
;  1758          1912  2      ! Copy key definition with defaults to block in key buffer.
;  1759          1913  2      !
;  1760          1914  2      CURRENT_KEY = KBF[KEY_KBF(.KEY_NUM-1)];
;  1761          1915  2      IF .CURRENT_KEY[KBF_ORDER_S] GEQ 0
;  1762          1916  2      THEN
;  1763          1917  2          CMD_ERROR(SOR$_KEYAMBINC);
;  1764          1918  2      CURRENT_KEY[KBF_ORDER] = .OPTS[B_DES];      ! Ascending/Descending
;  1765          1919  2      CURRENT_KEY[KBF_TYPE] = .TYPE;              ! DSC datatype
;  1766          1920  2      OPTS[B_DES] = FALSE;
;  1767          1921  2      OPTS[B_ASC] = FALSE;
;  1768          1922  2      CURRENT_KEY[KBF_POSITION] = .POSIT - 1;     ! Adjust to start at pos 0
;  1769          1923  2      CURRENT_KEY[KBF_LENGTH] = .SIZ;             ! Size
;  1770          1924  2      KBF[KEY_NUMBER] = MAXU(.KBF[KEY_NUMBER],  KEY_NUM);         ' Another key
;  1771          1925  2
;  1772          1926  2
;  1773          1927  2      ! We should have used up all the options
```

```
; 1774          1928  2      !
; 1775          1929  2      IF .OPTS NEQ 0 THEN CMD_ERROR(SOR$_KEYAMBINC);
; 1776          1930  2
; 1777          1931  1      END;


                                                   00676            .BLKB   2
                02  1C  1B  0B  0A  07  13  0E  06  06  00678 TAB:      .BYTE   6, 6, 14, 19, 7, 10, 11, 27, 28, 2     ;
                                                00  00682           .BYTE   0                                      ;
                                                01  00683           .BYTE   1                                      ;
                                                00  00684           .BYTE   0                                      ;
                                                15  00685           .BYTE   21                                     ;
                                                00  00686           .BYTE   0                                      ;
                                                05  00687           .BYTE   5                                      ;
                                                00  00688           .BYTE   0                                      ;
                                    14  04  03  00  00689           .BYTE   0, 3, 4, 20                            ;

                                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                47  4E  49  44  4E  45  43  53  41  09  0034C P.ACJ:  .ASCII  <9>\ASCENDING\                         ;
                            59  52  41  4E  49  42  06  00356 P.ACK:  .ASCII  <6>\BINARY\                            ;
                52  45  54  43  41  52  41  48  43  09  0035D P.ACL:  .ASCII  <9>\CHARACTER\                         ;
                        4C  41  4D  49  43  45  44  07  00367 P.ACM:  .ASCII  <7>\DECIMAL\                           ;
            47  4E  49  44  4E  45  43  53  45  44  0A  0036F P.ACN:  .ASCII  <10>\DESCENDING\                       ;
            47  4E  49  54  41  4F  4C  46  5F  46  0A  0037A P.ACO:  .ASCII  <10>\F_FLOATING\                       ;
            47  4E  49  54  41  4F  4C  46  5F  44  0A  00385 P.ACP:  .ASCII  <10>\D_FLOATING\                       ;
            47  4E  49  54  41  4F  4C  46  5F  47  0A  00390 P.ACQ:  .ASCII  <10>\G_FLOATING\                       ;
            47  4E  49  54  41  4F  4C  46  5F  48  0A  0039B P.ACR:  .ASCII  <10>\H_FLOATING\                       ;
    4E  47  49  53  5F  47  4E  49  44  41  45  4C  0C  003A6 P.ACS:  .ASCII  <12>\LEADING_SIGN\                     ;
                            52  45  42  4D  55  4E  06  003B3 P.ACT:  .ASCII  <6>\NUMBER\                            ;
49  53  5F  44  45  48  43  4E  55  50  52  45  56  4F  10  003BA P.ACU:  .ASCII  <16>\OVERPUNCHED_SIGN\               ;
                                                4E  47  003C9                                                        ;
                4E  4F  49  54  49  53  4F  50  08  003CB P.ACV:  .ASCII  <8>\POSITION\                          ;
4C  41  4D  49  43  45  44  5F  44  45  4B  43  41  50  0E  003D4 P.ACW:  .ASCII  <14>\PACKED_DECIMAL\             ;
                                            49  53  02  003E3 P.ACX:  .ASCII  <2>\SI\                                ;
                        44  45  4E  47  49  53  06  003E6 P.ACY:  .ASCII  <6>\SIGNED\                            ;
                                45  5A  49  53  04  003ED P.ACZ:  .ASCII  <4>\SIZE\                              ;
    4E  47  49  53  5F  45  54  41  52  41  50  45  53  0D  003F2 P.ADA:  .ASCII  <13>\SEPARATE_SIGN\            ;
    4E  47  49  53  5F  47  4E  49  4C  49  41  52  54  0D  00400 P.ADB:  .ASCII  <13>\TRAILING_SIGN\            ;
                    44  45  4E  47  49  53  4E  55  08  0040E P.ADC:  .ASCII  <8>\UNSIGNED\                          ;
                            44  45  4E  4F  5A  05  00417 P.ADD:  .ASCII  <5>\ZONED\                             ;
                                                   0041D            .BLKB   3
                                          00000015  00420           .LONG   21                                     ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00424 P.ACI:  .ADDRESS P.ACJ, P.ACK, P.ACL, P.ACM, P.ACN, -  ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 0043C                 P.ACO, P.ACP, P.ACQ, P.ACR, P.ACS, P.ACT, - ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00454                 P.ACU, P.ACV, P.ACW, P.ACX, P.ACY, P.ACZ, - ;
                              00000000' 00000000' 00000000' 0046C                 P.ADA, P.ADB, P.ADC, P.ADD            ;

                                                                    .PSECT  $OWN$,NOEXE,2

                                                00  00040 KEY_NUM:.BYTE   0                                      ;

                                                          KEYWORD_TABLE=      P.ACI

                                                                    .PSECT  $CODE$,NOWRT,2
```

```
                                01FC 00000 PARSE_KEY:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8                   ; 1586
                58      0000'   CF  9E 00002        MOVAB   KEY_NUM, R8
                57      F968    CF  9E 00007        MOVAB   CMD_ERROR, R7
                                68  96 0000C        INCB    KEY_NUM                                     ; 1673
                                54  94 0000E        CLRB    TYPE                                        ; 1674
                56              01  B0 00010        MOVW    #1, POSIT                                   ; 1675
                55              01  AE 00013        MNEGW   #1, SIZ                                     ; 1676
                                53  D4 00016        CLRL    OPTS                                        ; 1677
                        C0      A8  9F 00018 1$:    PUSHAB  STRING_DESC                                 ; 1679
                        0000'   CF  9F 0001B        PUSHAB  SD_KEY
        0000G   CF              02  FB 0001F        CALLS   #2, CLI_GET_VALUE
                03              50  E8 00024        BLBS    R0, 2$
                            012B    31 00027        BRW     22$
                        0000'   CF  9F 0002A 2$:    PUSHAB  KEYWORD_TABLE                               ; 1684
        0000V   CF              01  FB 0002E        CALLS   #1, LOOKUP_KEY
                52              50  D0 00033        MOVL    R0, LOOKUP
                0E              52  D1 00036        CMPL    LOOKUP, #14                                 ; 1685
                                1B  12 00039        BNEQ    4$
                        C0      A8  9F 0003B        PUSHAB  STRING_DESC                                 ; 1688
                                01  DD 0003E        PUSHL   #1
                    001C1108    8F  DD 00040        PUSHL   #1839368
                67              03  FB 00046        CALLS   #3, CMD_ERROR
                        C8      A8  B5 00049        TSTW    SUBVAL_DESC                                 ; 1689
                                05  13 0004C        BEQL    3$
                52              10  D0 0004E        MOVL    #16, LOOKUP                                 ; 1690
                                03  11 00051        BRB     4$
                52              0F  D0 00053 3$:    MOVL    #15, LOOKUP                                 ; 1691
50 00288000     8F              52  78 00056 4$:    ASHL    LOOKUP, #2654208, R0                        ; 1695
                                14  19 0005E        BLSS    5$
                        C8      A8  B5 00060        TSTW    SUBVAL_DESC                                 ; 1698
                                0F  13 00063        BEQL    5$
                        C0      A8  9F 00065        PUSHAB  STRING_DESC                                 ; 1700
                                01  DD 00068        PUSHL   #1
                    001C10FC    8F  DD 0006A        PUSHL   #1839356
                67              03  FB 00070        CALLS   #3, CMD_ERROR
                                04  00073        RET
50 77840800     8F              52  78 00074 5$:    ASHL    LOOKUP, #2005141504, R0                     ; 1708
                                15  18 0007C        BGEQ    8$
                                54  95 0007E        TSTB    TYPE                                        ; 1714
                                09  13 00080        BEQL    6$
                    001C8134    8F  DD 00082        PUSHL   #1868084
                67              01  FB 00088        CALLS   #1, CMD_ERROR
                54      FF5B CF42   90 0008B 6$:    MOVB    TAB[LOOKUP], TYPE                           ; 1718
                                85  11 00091 7$:    BRB     1$                                          ; 1706
50 88517000     8F              52  78 00093 8$:    ASHL    LOOKUP, #-2007928832, R0                    ; 1722
                                22  18 0009B        BGEQ    10$
                50      FF49 CF42   9A 0009D        MOVZBL  TAB[LOOKUP], R0                             ; 1732
00              53              50  E2 000A3        BBSS    R0, OPTS, 9$
                50      FF3F CF42   9A 000A7 9$:    MOVZBL  TAB[LOOKUP], R0                             ; 1733
                50              01  CC 000AD        XORL2   #1, R0
50              53              50  E5 000B0        BBCC    R0, OPTS, 14$
                    001C8134    8F  DD 000B4        PUSHL   #1868084                                    ; 1735
                67              01  FB 000BA        CALLS   #1, CMD_ERROR
                                45  11 000BD        BRB     14$                                         ; 1719
                0C              52  D1 000BF 10$:    CMPL    LOOKUP, #12                                 ; 1738
```

```
                                   1A 12 000C2          BNEQ    12$                          :
              09           53      08 E3 000C4          BBCS    #8, OPTS, 11$                : 1741
                     001C8134      8F DD 000C8          PUSHL   #1868084                     :
                           67      01 FB 000CE          CALLS   #1, CMD_ERROR                :
                              C8   A8 9F 000D1 11$:     PUSHAB  SUBVAL_DESC                  : 1742
                  0417     C7      01 FB 000D4          CALLS   #1, NUMERIC_PARSE            :
                           56      50 B0 000D9          MOVW    Z, POSIT                     :
                                   2B 11 000DC          BRB     16$                          :
                           10      52 D1 000DE 12$:     CMPL    LOOKUP, #16                  : 1745
                                   3B 12 000E1          BNEQ    17$                          :
              09           53      09 E3 000E3          BBCS    #9, OPTS, 13$                : 1750
                     001C8134      8F DD 000E7          PUSHL   #1868084                     :
                           67      01 FB 000ED          CALLS   #1, CMD_ERROR                :
                              C8   A8 9F 000F0 13$:     PUSHAB  SUBVAL_DESC                  : 1751
                  0417     C7      01 FB 000F3          CALLS   #1, NUMERIC_PARSE            :
              FFFFFFFF     8F      50 D1 000F8          CMPL    TMP, #-1                     : 1752
                                   05 12 000FF          BNEQ    15$                          :
                           55      01 AE 00101          MNEGW   #1, SIZ                      :
                                   8B 11 00104 14$:     BRB     7$                           :
                           55      50 B0 00106 15$:     MOVW    Z, SIZ                       :
   50         50           10      00 ED 00109 16$:     CMPZV   #0, #16, Z, Z                :
                                   81 13 0010E          BEQL    7$                           :
                                   7E D4 00110          CLRL    -(SP)                        :
                                   50 DD 00112          PUSHL   Z                            :
                                   02 DD 00114          PUSHL   #2                           :
                     001C102C      8F DD 00116          PUSHL   #1839148                     :
                                   33 11 0011C          BRB     21$                          :
                           0A      52 D1 0011E 17$:     CMPL    LOOKUP, #10                  : 1755
                                   03 13 00121          BEQL    19$                          :
                             FEF2  31 00123 18$:        BRW     1$                           :
              09           53      0A E3 00126 19$:     BBCS    #10, OPTS, 20$               : 1758
                     001C8134      8F DD 0012A          PUSHL   #1868084                     :
                           67      01 FB 00130          CALLS   #1, CMD_ERROR                :
                              C8   A8 9F 00133 20$:     PUSHAB  SUBVAL_DESC                  : 1759
                  0417     C7      01 FB 00136          CALLS   #1, NUMERIC_PARSE            :
                           68      50 90 0013B          MOVB    Z, KEY_NUM                   :
   50         50           08      00 ED 0013E          CMPZV   #0, #8, Z, Z                 :
                                   DE 13 00143          BEQL    18$                          :
                                   7E D4 00145          CLRL    -(SP)                        :
                                   50 DD 00147          PUSHL   Z                            :
                                   02 DD 00149          PUSHL   #2                           :
                     001C1014      8F DD 0014B          PUSHL   #1839124                     :
                           67      04 FB 00151 21$:     CALLS   #4, CMD_ERROR                :
                                      04 00154          RET                                  :
                           54      95 00155 22$:        TSTB    TYPE                         : 1798
                                   03 12 00157          BNEQ    23$                          :
                           54      0E 90 00159          MOVB    #14, TYPE                    :
              50 00300018  8F      54 78 0015C 23$:     ASHL    TYPE, #3145752, R0           : 1804
                                   23 18 00164          BGEQ    24$                          :
                           50      55 3C 00166          MOVZWL  SIZ, TMP                     : 1813
   51         54           03      00 EF 00169          EXTZV   #0, #3, TYPE, R1             : 1814
              51           01      51 78 0016E          ASHL    R1, #1, R1                   :
                           55      51 B0 00172          MOVW    R1, SIZ                      :
              10           53      09 E3 00175          BBCS    #9, OPTS, 24$                : 1818
   50         55           10      00 ED 00179          CMPZV   #0, #16, SIZ, TMP            : 1820
                                   09 13 0017E          BEQL    24$                          :
                     001C8034      8F DD 00180          PUSHL   #1867828                     :
```

```
                        67              01  FB 00186          CALLS   #1, CMD_ERROR
                    04  53              08  E5 00189 24$:     BBCC    #8, OPTS, 25$                   ;1834
                    09  53              09  E4 0018D          BBSC    #9, OPTS, 26$
                            001C8034    8F  DD 00191 25$:     PUSHL   #1867828                        ;1838
                        67              01  FB 00197          CALLS   #1, CMD_ERROR
                                        68  95 0019A 26$:     TSTB    KEY_NUM                         ;1843
                                        0C  12 0019C          BNEQ    27$
                            001C803C    8F  DD 0019E          PUSHL   #1867836                        ;1846
                        67              01  FB 001A4          CALLS   #1, CMD_ERROR
                        68              01  8E 001A7          MNEGB   #1, KEY_NUM                     ;1847
                        53      0400    8F  AA 001AA 27$:     BICW2   #1024, OPTS                     ;1849
                50 00320C18     8F          54  78 001AF      ASHL    TYPE, #3279896, R0              ;1855
                                            76  19 001B7      BLSS    39$
                        06                  54  91 001B9      CMPB    TYPE, #6                        ;1859
                                            4C  12 001BC      BNEQ    35$
                        01                  55  B1 001BE      CMPW    SIZ, #1                         ;1862
                                            31  13 001C1      BEQL    32$
                        02                  55  B1 001C3      CMPW    SIZ, #2                         ;1863
                                            05  12 001C6      BNEQ    28$
                        54                  07  90 001C8      MOVB    #7, TYPE
                                            27  11 001CB      BRB     32$
                        04                  55  B1 001CD 28$:     CMPW    SIZ, #4                     ;1864
                                            05  12 001D0      BNEQ    29$
                        54                  08  90 001D2      MOVB    #8, TYPE
                                            1D  11 001D5      BRB     32$
                        08                  55  B1 001D7 29$:     CMPW    SIZ, #8                     ;1865
                                            05  12 001DA      BNEQ    30$
                        54                  09  90 001DC      MOVB    #9, TYPE
                                            13  11 001DF      BRB     32$
                        10                  55  B1 001E1 30$:     CMPW    SIZ, #16                    ;1866
                                            05  12 001E4      BNEQ    31$
                        54                  1A  90 001E6      MOVB    #26, TYPE
                                            09  11 001E9      BRB     32$
                            001C8034    8F  DD 001EB 31$:     PUSHL   #1867828                        ;1867
                        67              01  FB 001F1          CALLS   #1, CMD_ERROR
                    0D  53              04  E1 001F4 32$:     BBC     #4, OPTS, 34$                   ;1868
                        1A                  54  91 001F8      CMPB    TYPE, #26                       ;1875
                                            05  12 001FB      BNEQ    33$
                        54                  19  90 001FD      MOVB    #25, TYPE                       ;1876
                                            03  11 00200      BRB     34$
                        54                  04  82 00202 33$:     SUBB2   #4, TYPE                    ;1877
                        53                  30  8A 00205 34$:     BICB2   #48, OPTS                   ;1880
                                            30  11 00208      BRB     41$                             ;1856
                        13                  54  91 0020A 35$:     CMPB    TYPE, #19                   ;1883
                                            22  12 0020D      BNEQ    40$
                    05  53                  04  E5 0020F      BBCC    #4, OPTS, 36$                   ;1886
                        54                  0F  90 00213      MOVB    #15, TYPE                       ;1888
                                            22  11 00216      BRB     41$
                        53                  20  8A 00218 36$:     BICB2   #32, OPTS                   ;1891
                    03  53                  02  E1 0021B      BBC     #2, OPTS, 37$                   ;1892
                        54                  11  90 0021F      MOVB    #17, TYPE
                        53                  0C  8A 00222 37$:     BICB2   #12, OPTS                   ;1894
                        04                  53  E9 00225      BLBC    OPTS, 38$                       ;1895
                                            54  97 00228      DECB    TYPE                            ;1900
                                            55  B6 0022A      INCW    SIZ                             ;1901
                        53                  03  8A 0022C 38$:     BICB2   #3, OPTS                    ;1904
                                            09  11 0022F 39$:     BRB     41$                         ;1885
```

```
                                                001C1124   8F   DD 00231 40$:    PUSHL   #1839396                                ; 1909
                                             67            01   FB 00237         CALLS   #1, CMD_ERROR
                                             50            68   9A 0023A 41$:    MOVZBL  KEY_NUM, R0                            ; 1914
                                             50            02   C4 0023D         MULL2   #2, R0
                                             52       E0 B840   DE 00240         MOVAL   @KBF[R0], CURRENT_KEY
                                             52            06   C2 00245         SUBL2   #6, CURRENT_KEY
                                                      02   A2   B5 00248         TSTW    2(CURRENT_KEY)                         ; 1915
                                                           09   19 0024B         BLSS    42$
                                                001C8134   8F   DD 0024D         PUSHL   #1868084                               ; 1917
                                             67            01   FB 00253         CALLS   #1, CMD_ERROR
                                50                    53   01   07   EF 00256 42$:    EXTZV   #7, #1, OPTS, R0               ; 1918
                                                      02   A2   50   B0 0025B         MOVW    R0, 2(CURRENT_KEY)
                                                           62   54   9B 0025F         MOVZBW  TYPE, (CURRENT_KEY)            ; 1919
                                                           53   C0   8F   8A 00262    BICB2   #192, OPTS                     ; 1921
                                                      04   A2   56   01   A3 00266    SUBW3   #1, POSIT, 4(CURRENT_KEY)      ; 1922
                                                      06   A2   55   B0   0026B       MOVW    SIZ, 6(CURRENT_KEY)            ; 1923
                                                           50   E0   B8   3C 0026F    MOVZWL  @KBF, R0                       ; 1924
                                50                    68   08   00   ED   00273      CMPZV   #0, #8, KEY_NUM, R0
                                                           03   1B   00278         BLEQU   43$
                                                           50   68   9A 0027A       MOVZBL  KEY_NUM, R0
                                                      E0   B8   50   B0   0027D 43$:    MOVW    R0, @KBF
                                                           53   D5   00281         TSTL    OPTS                           ; 1929
                                                           09   13   00283         BEQL    44$
                                                001C8134   8F   DD 00285         PUSHL   #1868084
                                             67            01   FB 0028B         CALLS   #1, CMD_ERROR
                                                           04   0028E 44$:       RET                                    ; 1931

; Routine Size:  655 bytes,    Routine Base:  $CODE$ + 068D

; 1778          1932  1
; 1779          1933  1 ROUTINE SET_KEY: NOVALUE =
; 1780          1934  2     BEGIN
; 1781          1935  2     WHILE CLI_NEXT_QUAL(SD_KEY) DO PARSE_KEY(); ! Do it again and again
; 1782          1936  1     END;

                                                      0000 00000 SET_KEY:.WORD   Save nothing                                   ; 1933
                                                0000'  CF   9F   00002 1$:      PUSHAB  SD_KEY                                 ; 1935
                                          0000G  CF            01   FB 00006    CALLS   #1, CLI_NEXT_QUAL
                                                 07            50   E9 0000B    BLBC    R0, 2$
                                          FD5E   CF            00   FB 0000E    CALLS   #0, PARSE_KEY
                                                               ED   11 00013    BRB     1$
                                                                    04 00015 2$:    RET                                    ; 1936

; Routine Size:  22 bytes,    Routine Base:  $CODE$ + 091C
```

```
;  1784          1937  1 ROUTINE SET_COLL: NOVALUE =
;  1785          1938  1
;  1786          1939  1 !++
;  1787          1940  1 !
;  1788          1941  1 ! Functional Description:
;  1789          1942  1 !
;  1790          1943  1 !       Routine to call CLI to get the argument of the /COLLATING_SEQUE
;  1791          1944  1 !       qualifier.  The valid arguments are ASCII, EBCDIC and MULTINATIONAL.
;  1792          1945  1 !       If no argument is given, ASCII is assumed.
;  1793          1946  1 !
;  1794          1947  1 ! Formal Parameters:
;  1795          1948  1 !
;  1796          1949  1 !       QUAL_DESC       address of the qualifier descriptor block.
;  1797          1950  1 !
;  1798          1951  1 ! Implicit Outputs:
;  1799          1952  1 !
;  1800          1953  1 !       OPT is modified to have the appropriate option bit set.
;  1801          1954  1 !
;  1802          1955  1 !--
;  1803          1956  2     BEGIN
;  1804          1957  2
;  1805          1958  2     KEY_TABLE_(STR_OPT_COLL);
;  1806          1959  2
;  1807          1960  2     OWN
;  1808          1961  2         TAB: VECTOR[TABLE_SIZE,BYTE]
;  1809          1962  2             PSECT($CODE$) PRESET(
;  1810          1963  2                 [K_ASCI]=       0,
;  1811          1964  2                 [K_EBCD]=       SOR$M_EBCDIC,
;  1812          1965  2                 [K_MULT]=       SOR$M_MULTI);
;  1813          1966  2     LOCAL
;  1814          1967  2         LOOKUP;
;  1815          1968  2
;  1816          1969  2     CLI_GET_VALUE( SD_COLL, STRING_DESC[BASE_] );
;  1817          1970  2     LOOKUP = LOOKUP_KEY(KEYWORD_TABLE);
;  1818          1971  2     IF .LOOKUP LSSU TABLE_SIZE THEN OPT = .OPT OR .TAB[.LOOKUP];
;  1819          1972  2
;  1820          1973  1     END;


                                                        00932          .BLKB   2
                                                    00  00934 TAB:     .BYTE   0                                       ;
                                                00G 00* 00935          .BYTE   <SOR$M_EBCDIC&255>, SOR$M_MULTI         ;

                                                                       .PSECT  $PLIT$,NOWRT,NOEXE,2

                                            49  49  43  53  41  05  00478 P.ADF:  .ASCII  <5>\ASCII\                      ;
                                        43  49  44  43  42  45  06  0047E P.ADG:  .ASCII  <6>\EBCDIC\                     ;
    4C  41  4E  4F  49  54  41  4E  49  54  4C  55  4D  0D  00485 P.ADH:  .ASCII  <13>\MULTINATIONAL\                   ;
                                                            00493          .BLKB   1
                                                    00000003    00494          .LONG   3                               ;
                                    00000000' 00000000' 00000000' 00498 P.ADE:  .ADDRESS P.ADF, P.ADG, P.ADH            ;

                                                                  KEYWORD_TABLE=    P.ADE


                                                                       .PSECT  $CODE$,NOWRT,2
```

```
                                 0000 00000 SET_COLL:
                                                    .WORD   Save nothing                    ; 1937
                            0000'  CF  9F 00002             PUSHAB  STRING_DESC             ; 1969
                            0000'  CF  9F 00006             PUSHAB  SD_COLL                 ;
                     0000G  CF         02  FB 0000A         CALLS   #2, CLI_GET_VALUE       ;
                            0000'  CF  9F 0000F             PUSHAB  KEYWORD_TABLE           ; 1970
                     0000V  CF         01  FB 00013         CALLS   #1, LOOKUP_KEY          ;
                            03         50  D1 00018         CMPL    LOOKUP, #3              ; 1971
                                       0A  1E 0001B         BGEQU   1$                      ;
                            51   DC AF40  9A 0001D         MOVZBL  TAB[LOOKUP], R1         ;
                     0000'  CF         51  C8 00022         BISL2   R1, OPT                 ;
                                       04  00027 1$:        RET                             ; 1973

; Routine Size:  40 bytes,    Routine Base:  $CODE$ + 0937
```

```
;  1822          1974  1 ROUTINE LOOKUP_KEY
;  1823          1975  1         (
;  1824          1976  1         TABLE:  REF VECTOR
;  1825          1977  1         ) =
;  1826          1978  1
;  1827          1979  1 !++
;  1828          1980  1 !
;  1829          1981  1 ! FUNCTIONAL DESCRIPTION:
;  1830          1982  1 !
;  1831          1983  1 !       This routine is called to process the qualifier value for qualifiers
;  1832          1984  1 !       that take keyword values.
;  1833          1985  1 !
;  1834          1986  1 ! FORMAL PARAMETERS:
;  1835          1987  1 !
;  1836          1988  1 !       TABLE   - Pointer to keyword table
;  1837          1989  1 !
;  1838          1990  1 ! IMPLICIT INPUTS:
;  1839          1991  1 !
;  1840          1992  1 !       STRING_DESC contains the keyword to match against, and possibly a value.
;  1841          1993  1 !       IOS_CMDFLAGS has already been initialized to default values.
;  1842          1994  1 !
;  1843          1995  1 ! IMPLICIT OUTPUTS:
;  1844          1996  1 !
;  1845          1997  1 !       SUBVAL_DESC is set to describe the value (if any) on this keyword.
;  1846          1998  1 !
;  1847          1999  1 ! ROUTINE VALUE:
;  1848          2000  1 !
;  1849          2001  1 !       If the keyword was found, the index of the keyword, otherwise -1.
;  1850          2002  1 !
;  1851          2003  1 ! SIDE EFFECTS:
;  1852          2004  1 !
;  1853          2005  1 !       If the keyword was not found, a diagnostic is issued.
;  1854          2006  1 !
;  1855          2007  1 !--
;  1856          2008  1
;  1857          2009  2     BEGIN
;  1858          2010  2     MAP
;  1859          2011  2         TABLE:  REF VECTOR;
;  1860          2012  2
;  1861          2013  2     LOCAL
;  1862          2014  2         L,                                  ! Keyword string length
;  1863          2015  2         A:      REF VECTOR[,BYTE],          ! Keyword string address
;  1864          2016  2         INDEX;                              ! Routine return value
;  1865          2017  2
;  1866          2018  2
;  1867          2019  2     ! Search through table looking for a single match. If more than one
;  1868          2020  2     ! match is found return an error.
;  1869          2021  2     !
;  1870          2022  2     INDEX = -1;                                     ! Initialize return to error
;  1871          2023  2
;  1872          2024  2     ! Remove the value (if any) from this keyword
;  1873          2025  2     !
;  1874          2026  2     A = .STRING_DESC[DSC$A_POINTER];
;  1875          2027  2     L = .STRING_DESC[DSC$W_LENGTH];
;  1876          2028  2     L = .L - KEYVALU();
;  1877          2029  2     L = .L - .SUBVAL_DESC[DSC$W_LENGTH];
;  1878          2030  2
```

```
;1879     2031  2     INCR N FROM 0 TO .TABLE[-1]-1 DO
;1880     2032  3         BEGIN
;1881     2033  3         LOCAL
;1882     2034  3             P:  REF VECTOR[,BYTE];
;1883     2035  3         P = .TABLE[.N];                     ! Point to ASCIC keyword string
;1884     2036  3         IF .P[0] GEQ .L<0,8> THEN
;1885     2037  3         IF CH$EQL(.L, A[0], .L, P[1])
;1886     2038  3         THEN
;1887     2039  4             BEGIN
;1888     2040  4             !
;1889     2041  4             ! We found a matching name.
;1890     2042  4             !
;1891     2043  4             IF .INDEX LSS 0
;1892     2044  4             THEN
;1893     2045  4                 INDEX = .N
;1894     2046  4             ELSE
;1895     2047  5                 BEGIN                                                                   1
;1896     2048  5                 CMD_ERROR(SOR$_SHR_BADKEY, 1, STRING_DESC[BASE_]);                      1
;1897     2049  5                 RETURN -1;
;1898     2050  4                 END;
;1899     2051  4             IF .P[0] EQL .L<0,8> THEN EXITLOOP; ! An optimization
;1900     2052  3             END;
;1901     2053  2         END;
;1902     2054  2
;1903     2055  2     IF .INDEX LSS 0
;1904     2056  2     THEN
;1905     2057  3         BEGIN
;1906     2058  3         CMD_ERROR(SOR$_SHR_BADKEY, 1, STRING_DESC[BASE_]);
;1907     2059  3         RETURN -1;
;1908     2060  2         END;
;1909     2061  2
;1910     2062  2     RETURN .INDEX;                          ! Return value
;1911     2063  1     END;


                                        07FC 00000 LOOKUP_KEY:
                                                   .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10        ; 1974
                                5A      0000'  CF  9E 00002    MOVAB    STRING_DESC, R10            ;
                                58             CE  CE 00007    MNEGL    #1, INDEX                   ; 2022
                                59        04   AA  D0 0000A    MOVL     STRING_DESC+4, A            ; 2026
                                56             6A  3C 0000E    MOVZWL   STRING_DESC, L              ; 2027
                          FA59  CF             00  FB 00011    CALLS    #0, KEYVALU                 ; 2028
                                56             50  C2 00016    SUBL2    R0, L                       ;
                                50        08   AA  3C 00019    MOVZWL   SUBVAL_DESC, R0             ; 2029
                                56             50  C2 0001D    SUBL2    R0, L                       ;
                                57        04   AC  D0 00020    MOVL     TABLE, R7                   ; 2031
                                55             01  CE 00024    MNEGL    #1, N                       ; 2036
                                               1C  11 00027    BRB      2$                          ;
                                54           6745  D0 00029 1$: MOVL     (R7)[N], P                  ; 2035
                                56             64  91 0002D    CMPB     (P), L                      ; 2036
                                               13  1F 00030    BLSSU    2$                          ;
                          01    A4       69      56  29 00032    CMPC3    L, (A), 1(P)                ; 2037
                                               0C  12 00037    BNEQ     2$                          ;
                                               58  D5 00039    TSTL     INDEX                       ; 2043
```

```
                                        11  18 0003B          BGEQ    4$                                          ;
                                58      55  D0 0003D          MOVL    N, INDEX                                    ; 2045
                                56      64  91 00040          CMPB    (R), L                                      ; 2051
                                        05  13 00043          BEQL    3$                                          ;
                      DF        55   FC   A7  F2 00045 2$:     AOBLSS  -4(R7), N, 1$                               ; 2031
                                        58  D5 0004A 3$:      TSTL    INDEX                                       ; 2055
                                        13  18 0004C          BGEQ    5$                                          ;
                                        5A  DD 0004E 4$:      PUSHL   R10                                         ; 2058
                                        01  DD 00050          PUSHL   #1                                          ;
                                001C110C    8F  DD 00052      PUSHL   #1839372                                    ;
                            F644    CF      03  FB 00058      CALLS   #3, CMD_ERROR                               ;
                                50          01  CE 0005D      MNEGL   #1, R0                                      ; 2059
                                            04  00060        RET                                                 ;
                                50          58  D0 00061 5$:  MOVL    INDEX, R0                                   ; 2062
                                            04  00064        RET                                                 ; 2063

; Routine Size:  101 bytes,    Routine Base:  $CODE$ + 095F

; 1912          2064  1
; 1913          2065  1 END
; 1914          2066  0 ELUDOM



;                                   PSECT SUMMARY
;
;      Name                     Bytes                          Attributes
;
; $PLIT$                        1188  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
; $OWN$                           65  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
; $CODE$                        2500  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                               Library Statistics
;
;                                        -------- Symbols --------      Pages       Processing
;      File                              Total    Loaded   Percent      Mapped      Time
;
; _$255$DUA28:[SYSLIB]LIB.L32;1          18619        51         0      1000          00:01.7



;                                       COMMAND QUALIFIERS

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

```
; Size:          2468 code + 1285 data bytes
; Run Time:        00:52.8
; Elapsed Time:    02:45.5
; Lines/CPU Min:    2349
; Lexemes/CPU-Min: 24175
; Memory Used:  333 pages
; Compilation Complete
```



```
ADT
B
BLV
BPV
BU
CIT
CMP
CMP
CMP
COM
COM
D
DC
DSC
DTY
F
FC
FLD
FLD
G
GC
H
HC
JOI
L
LIB
LIB
LIB
LIB
LIB
LIB
LIB
LU
M1
NL
NLC
NR
NRC
NU
NZ
O
OU
O_1
P
P1
Q
QU
SOR
SOR
SOR
T
TWC
V
VAC
VB
VBL
VBF
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SORCOMMAN
LIS

LIBFIXUPD
LIS

REQSYM
R32

CRETRANS
LIS

SFKEYWRD
LIS

SORCOLLAT
LIS

OPCODES
LIS

SORARCHAI
LIS

SORCOLUTI
LIS

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SORINTERF
LIS

SORKEYSUB
LIS

SORFILNAM
LIS

SORCOMPAR
LIS

SORENTRY
LIS