Table of contents

```
                                .TITLE  SORTO   RTSORT -- ROOT MODULE
                                .ENABL  LC
                        ;
                        ;
                        ;  Copyright (c) 1977,1978,1979,1980.
                        ;  S&H Computer Systems Inc.
                        ;  Nashville, Tennessee
                        ;
000000                          .CSECT  SORTO
000000'                 POBASE  =       .
                                .ENABL  LC
                                .SBTTL  DATA AREAS
                        ;
                        ;  ROOT MODULE OF RT-11/TSX SORT PROGRAM.
                        ;
                        ;  WRITTEN BY PHIL SHERROD.
                        ;
                        ;
                        ;  LINK COMMAND TO FORM OVERLAYED VERSION OF RTSORT:
                        ;
                        ;  .R LINK
                        ;  *RTSORT=SORTO/C
                        ;  *SORTI/O:1/C
                        ;  *SORT1/O:1/C
                        ;  *SORT2/O:1
                        ;
                        ;
                        ;###############################################################################
                        ;
                        ; Modified in Feburary, 1980 by K.C. Lidster of DISC to also execute
                        ; under the RT11 run-time for the RSTS operating system.  This mainly
                        ; involved the handling of the extra file parameters pertaining to
                        ; RSTS (PPN, clustersize, etc.).
                        ;
                        ;###############################################################################
                        ;
                        ;
                        ;  GLOBAL DEFINITIONS
                        ;
                                .GLOBL  QBLK,QLEN,MODE,TSXFL
                                .GLOBL  TOPMEM,NUMKY2,RECLEN,ISMARS
                                .GLOBL  SKPCNT,MXREC1,MXREC2
                                .GLOBL  EOFCHR,EOFCOL
                                .GLOBL  INFLEN,KYUCA,EOFBUF
                                .GLOBL  BLKFC,OWCNT,OBUFSZ,START1
                                .GLOBL  STACK,P1TERM,P2TERM,KTYPE
                                .GLOBL  KAD,KSTRT,KLEN,OBUF1,OBUF2
                                .GLOBL  NXTBLK,OBFEND,ORPOS,FLBLK
                                .GLOBL  ITYPE,OTYPE,F$INPL
                                .GLOBL  RT$CBL,RT$FTN,RT$DBL,RT$TXT,RT$X
                                .GLOBL  RT$OS,RT$OR,RT$OX,RT$F11
                                .GLOBL  RT$RA,RT$RB,CURSAV,ENDSAV,SAVSPC,SKPVAL
                                .GLOBL  RT$VLN,RT$CR,MULTPY,DIVIDE,SRTCNT,TMPCNT
                                .GLOBL  CISRTI,CISRT1,CISRT2
                                .GLOBL  NUMBUF,NUMCOL,NUMLEN
                                .GLOBL  IFPNT,CUROFL,LOWMEM,RWDLEN
                                .GLOBL  RCNT1,RCNT2,FLUSH,OUTWRT,MXFILS
```

```
                                        .GLOBL  MXFIL2,COMPAR,ABORT,FILFL
                                        .GLOBL  FILNB,FILRN,FILST,MSGNAM
                                        .GLOBL  REC1,REC2,KYDIS
                                        .GLOBL  T1SIZ,T2SIZ,DSKCVT
                                        .GLOBL  JSW,OVLBIT,POTOP,POBASE
                                        .GLOBL  UEOF,V3FLAG,ANDFLG,ORFLG
                                        .GLOBL  KYCOMP,INRLEN,SELADR,FLDMAP,MAPEND
                                        .GLOBL  F$BIN,F$CR,F$PAUS,F$VLEN
                                        .GLOBL  KYBIN,KYLSS,KYTSS,KYDATE,SFLAGS
                                        .GLOBL  SETNRT,RSTS,CHNPPN,FILPPN,SPCFLG
                                ;
                                ;  GLOBAL REFERENCES
                                ;
                                        .GLOBL  SORTP1,SORTP2,TERM
                                        .GLOBL  SRTINI,P1TOP,P2TOP
                                ;
                                ;  PARAMETERS
                                ;
000310                          MXFILS  =       200.            ;MAX # OF TEMP FILES
000620                          MXFIL2  =       2*MXFILS
000017                          MXKEYS  =       15.             ;MAX # OF SORT KEYS PER RECORD
000012                          LF      =       12              ;ASCII LINE FEED
000007                          BELL    =       07              ;ASCII BELL

                                .IF DF RT11
                                        ;
                                        ;  SET INITIAL STACK POINTER
                                                .ASECT
000042                                  .       =       42
000042  003336'                                 .WORD   STACK
000000                                          .CSECT  SORTO
                                .ENDC

                                ;
                                ;  General control flags stored in SFLAGS:
                                ;
000001                          F$BIN   =       1       ;Records contain binary data
000002                          F$VLEN  =       2       ;Records are variable length
000004                          F$CR    =       4       ;Insert CR-LF at end of output records
000010                          F$PAUS  =       10      ;Pause on completion of reading input file
000020                          F$INPL  =       20      ;Do an inplace sort
                                ;
                                ;  File type flags stored in ITYPE & OTYPE.
                                ;
000001                          RT$CBL  =       1       ;COBOL file
000002                          RT$DBL  =       2       ;DIBOL file
000004                          RT$FTN  =       4       ;FORTRAN file
000010                          RT$F11  =       10      ;FILES-11 file
000020                          RT$TXT  =       20      ;TEXT file
000040                          RT$X    =       40      ;General fixed length records
000100                          RT$OS   =       100     ;Sequential organization
000200                          RT$OR   =       200     ;Relative organization
000400                          RT$OX   =       400     ;Indexed organization
001000                          RT$RA   =       1000    ;Ascii recording mode
002000                          RT$RB   =       2000    ;Binary recording mode
040000                          RT$CR   =       40000   ;Carriage-return is record terminator
100000                          RT$VLN  =       100000  ;Blank strip the records
```

DATA AREAS

```
                                ;
                                ;  FILE MANAGEMENT TABLES
                                ;
000000                          FILST:  .BLKW   MXFILS          ;FILE STARTING BLOCK NUMBER
000620                          FILNB:  .BLKW   MXFILS          ;# OF BLOCKS IN FILE
001440                          FILRN:  .BLKW   MXFILS          ;# OF SORTED RUNS IN FILE
                                ;
                                ;  SORT KEY DESCRIPTOR TABLES
                                ;
002260                          KTYPE:  .BLKW   MXKEYS          ;KEY TYPE
002316                          KAD:    .BLKW   MXKEYS          ;0=>ASCENDING; 1=>DESCENDING
002354                          KSTRT:  .BLKW   MXKEYS          ;KEY STARTING COLUMN
002412                          KLEN:   .BLKW   MXKEYS          ;KEY LENGTH (COLUMNS)
                                ;
                                ;  DATA AREAS
                                ;
002450  000000                  SFLAGS: .WORD   0
002452  000000                  ITYPE:  .WORD   0
002454  000000                  OTYPE:  .WORD   0
002456  000000                  NUMCOL: .WORD   0
002460  000000                  NUMLEN: .WORD   0


002462  000000                  CUROFL: .WORD   0
002464  000000                  TOPMEM: .WORD   0
002466  000000                  COMPAR: .WORD   0
002470                          MSGNAM: .BLKW   3
002476  000000                  NUMKY2: .WORD   0
002500  000000                  RECLEN: .WORD   0
002502  000000                  ISMARS: .WORD   0
002504  000000                  RWDLEN: .WORD   0
002506  000000                  RCNT1:  .WORD   0
002510  000000                  RCNT2:  .WORD   0
002512  000000                  LOWMEM: .WORD   0
002514  000000                  SKPCNT: .WORD   0
002516  000000                  SKPVAL: .WORD   0
002520  000000                  MXREC1: .WORD   0
002522  000000                  MXREC2: .WORD   0
002524  000000                  T1SIZ:  .WORD   0
002526  000000                  T2SIZ:  .WORD   0
002530  000000                  INFLEN: .WORD   0
002532  000000  000000          SRTCNT: .WORD   0,0
002536  000000  000000          TMPCNT: .WORD   0,0
002542                          AREA:   .BLKW   10.
002566  000000                  OWCNT:  .WORD   0
002570  000000                  OBUFSZ: .WORD   0
002572  000000                  REC1:   .WORD   0
002574  000000                  REC2:   .WORD   0
002576  000000                  OBUF1:  .WORD   0
002600  000000                  OBUF2:  .WORD   0
002602  000000                  NXTBLK: .WORD   0
002604  000000                  FILFL:  .WORD   0
002606  000000                  OBFEND: .WORD   0
002610  000000                  ORPOS:  .WORD   0
002612  000000                  FLBLK:  .WORD   0
```

DATA AREAS

```
                                BLKFC:  .WORD   0
                                IFPNT:  .WORD   0
                                EOFBUF: .WORD   0
                                FLDMAP: .WORD   0
                                MAPEND: .WORD   0
                                SELADR: .WORD   0
                                INRLEN: .WORD   0
                                EOFCOL: .WORD   0
                                CHRCNT: .WORD   0
                                CURSAV: .WORD   0
                                ENDSAV: .WORD   0
                                SAVSPC: .BLKW   30.
                                ;
                                ;  DEFINE STACK SPACE
                                        .BLKW   178.
                                STACK:
                                ;
                                ;  BYTE DATA


                                EOFCHR: .BYTE   0
                                UEOF:   .BYTE   0
                                ANDFLG: .BYTE   0
                                ORFLG:  .BYTE   0
                                SC1:    .BYTE   0
                                SC2:    .BYTE   0
                                SPCFLG: .BYTE   0
                                NUMBUF: .ASCII  /12345678/
                                        .LIST   BIN
PAUSMS: .ASCIZ  /SORT input complete.  Mount SORT output disk./<BELL>
                                        .LIST   BIN
                                        .EVEN
                                ;
                                ;  Table to convert non-separate sign characters for COBOL and DIBOL
                                ;  signed numeric display values into the natural digit and a
                                ;  sign indication.
                                ;
                                        .NLIST  BEX
                        =               .-60            ;First char in table is "0" (octal 60).
                                SINCHR  .BYTE   +<'0>,+<'1>,+<'2>,+<'3>,+<'4>   ; 060 - 064
                                        .BYTE   +<'5>,+<'6>,+<'7>,+<'8>,+<'9>   ; 065 - 071
                                        .BYTE       0,    0,    0,    0,    0   ; 072 - 076
                                        .BYTE       0,    0,+<'1>,+<'2>,+<'3>   ; 077 - 103
                                        .BYTE   +<'4>,+<'5>,+<'6>,+<'7>,+<'8>   ; 104 - 110
                                        .BYTE   +<'9>,-<'1>,-<'2>,-<'3>,-<'4>   ; 111 - 115
                                        .BYTE   -<'5>,-<'6>,-<'7>,-<'8>,-<'9>   ; 116 - 122
                                        .BYTE       0,    0,    0,    0,    0   ; 123 - 127
                                        .BYTE       0,    0,    0,    0,    0   ; 130 - 134
                                        .BYTE       0,    0,    0,    0,    0   ; 135 - 141
                                        .BYTE       0,    0,    0,    0,    0   ; 142 - 146
                                        .BYTE       0,    0,    0,    0,    0   ; 147 - 153
                                        .BYTE       0,    0,    0,    0,-<'0>   ; 154 - 160
                                        .BYTE   -<'1>,-<'2>,-<'3>,-<'4>,-<'5>   ; 161 - 165
                                        .BYTE   -<'6>,-<'7>,-<'8>,-<'9>,    0   ; 166 - 172
                                        .BYTE   +<'0>,    0,-<'0>               ; 173 - 175
```

```
                                        .EVEN
                                        .LIST   BEX

                                ;
                                ; The following macro call generates hidden system dependent data
                                ;
                                ;
003052                                  $system


                                ;
                                ;  Define file descriptors and file specification blocks
                                ;
004264                          fd1::   $FDB            ;file descriptor blocks
004274                          fd2::   $FDB
004304                          fd3::   $FDB
004314                          fd4::   $FDB

004324                          spc1::  $SPC            ;file specification blocks (system dependent names)
004336                          spc2::  $SPC
004350                          spc3::  $SPC
004362                          spc4::  $SPC
004374                          spc5::  $SPC
004406                          spc6::  $SPC
004420                          spc7::  $SPC
004432                          spc8::  $SPC
004444                          spc9::  $SPC

                                ; table of pointers to SPC's
004456  004324' 004336' 004350' files:: .word   spc1,   spc2,   spc3,   spc4,   spc5
004464  004362' 004374'
004470  004406' 004420' 004432'         .word   spc6,   spc7,   spc8,   spc9,   0
004476  004444' 000000
                                ;
                                ;  MACRO TO ABORT SORT
                                ;
                                        .MACRO  XXX     COD
                                XXX'COD =       .
                                        MOV     #'COD'.,R0
                                        CALL    ABORT
                                        .ENDM   XXX

                                ;-------------------------------------------------------------------------------
                                ;  Debugging breakpoint checkins.
                                ;
                                ;  Checkin from SORT1.
                                ;
004502  000207                 CISRT1: RETURN
                                ;
                                ;  Checkin from SORT1.
                                ;
004504  000207                 CISRT1: RETURN
                                ;
                                ;  Checkin from SORT2.
                                ;
004506  000207                 CISRT2: RETURN
```

```
                                        .SBTTL  MAIN SORT FLOW
                                ;-------------------------------------------------------
                                ;
004510                          START:
                                ;
                                ;  ENTER SORTI TO DO INITIALIZATION
                                ;
004510  000167  000000G                 JMP     SRTINI          ;ENTER INITIALIZATION ROUTINE
                                ;
                                ;  SORTI JUMPS TO START1 WHEN IT IS READY TO START A SORT
                                ;
004514  000167  000000G         START1: JMP     SORTP1          ;BEGIN SORT PHASE 1
                                ;
                                ;  TERMINATION OF PHASE 1
                                ;
004520                          P1TERM: $LOCK                   ;LOCK USR MODULE IN MEMORY FOR SPEED
004522                                  $KEEP   #FD2            ;CLOSE TEMP FILE 1 CHANNEL
                                ;  CLOSE TEMP FILE # 2 AND DELETE IT IF IT WAS NOT USED.
004570  005767  175732                  TST     T2SIZ           ;DID WE OPEN TEMP FILE 2?
004574  001503                          BEQ     7$              ;BR IF NOT
004576                                  $KEEP   #FD4            ;CLOSE TEMP FILE 2 CHANNEL
004644  016702  175732                  MOV     NXTBLK,R2       ;GET LAST CLUSTER #
004650  004767  002022                  CALL    DSKCVT          ;GET FILE # FOR IT
004654  020427  004314'                 CMP     R4,#FD4         ;IS TEMP FILE 2 NEEDED?
004660  001451                          BEQ     7$              ;BR IF YES
                                ;  TEMP FILE 2 IS NOT NEEDED SO DELETE IT
004662  005067  175640                  CLR     T2SIZ           ;SAY FILE NOT USED
004666                                  $ERASE  #FD4,#SPC3
                                ;
                                ;  SEE IF WE SHOULD PAUSE WHILE OUTPUT DISK IS MOUNTED.
                                ;
005004  032767  000010  175436  7$:     BIT     #F$PAUS,SFLAGS  ;IS PAUSE WANTED?
005012  001412                          BEQ     1$              ;BR IF NOT
005014                                  $UNLOCK                 ;UNLOCK USR MODULE DURING PAUSE
005016                                  $PRINT  #PAUSMS         ;PRINT PAUSE MESSAGE
005024                          5$:     .TTYIN                  ;WAIT FOR USER TO ENTER CR/LF
005030  120027  000012                  CMPB    R0,#LF
005034  001373                          BNE     5$              ;LOOP TILL LF ENTERED
005036                                  $LOCK                   ;RELOCK USR MODULE
                                ;
                                ;  SEE IF AN IN-PLACE SORT IS WANTED.
                                ;
005040  032767  000020  175402  1$:     BIT     #F$INPL,SFLAGS  ;WANT IN-PLACE SORT?
005046  001435                          BEQ     4$              ;BR IF NOT
                                ;  IN-PLACE SORT IS WANTED.  DO LOOKUP ON INPUT FILE FOR OUTPUT.
005050                                  $REOPEN #FD3,#SPC4      ;DO REOPEN TO ASSOCIATE OUTPUT WITH INPUT FILE
005136  000167  000510                  JMP     OPNTMP
                                ;  THIS IS NOT AN IN-PLACE SORT.
                                ;  IF OUTPUT FILE EXISTS NOW DELETE IT.
005142                          4$:     $ERASE  #FD3,#SPC1
                                ;  TRY TO OPEN NEW OUTPUT FILE
                                ;
                                ;  Determine how many blocks to request for output file.
                                ;
005260  032767  100000  175166  TONOF:  BIT     #RT$VLN,OTYPE   ;ARE OUTPUT RECORDS BLANK STRIPPED?
005266  001033                          BNE     4$              ;BR IF YES
                                ;  Fixed length output records.
```

```
                                        ;  Multiply record size by number of records to determine needed size.
005270  016704  175236                          MOV     SRTCNT,R4       ;GET # OF RECORDS PASSED TO WORD (LOW-ORDER)
005274  016705  175234                          MOV     SRTCNT+2,R5     ;GET HIGH-ORDER COUNT
005300  016700  175174                          MOV     RECLEN,R0       ;GET OUTPUT RECORD LENGTH
005304  032767  040000  175142                  BIT     #RT$CR,OTYPE    ;ARE WE TO ADD CR-LF TO OUTPUT RECORDS?
005312  001010                                  BNE     5$              ;BR IF YES
005314  032767  000001  175132                  BIT     #RT$CBL,OTYPE   ;IS THIS A COBOL FILE?
005322  001406                                  BEQ     6$              ;BR IF NOT
005324  032767  002000  175122                  BIT     #RT$RB,OTYPE    ;COBOL BINARY FILE?
005332  001402                                  BEQ     6$              ;BR IF NOT
005334  062700  000002                  5$:     ADD     #2,R0           ;WE NEED 2 MORE CHARS IN RECORD SIZE
005340  004767  002112                  6$:     CALL    MULTPY          ;MULTIPLY # RECORDS TIMES RECORD SIZE
005344  012700  001000                          MOV     #512.,R0        ;NOW DIVIDE BY # CHARS PER DISK BLOCK
005350  004767  002154                          CALL    DIVIDE
005354  000436                                  BR      7$
                                        ;  Variable length (blank stripped) output records.
                                        ;  Use size of temp file to detmine space needed for output file.
005356  016704  175154                  4$:     MOV     TMPCNT,R4       ;GET # DATA BYTES WRITTEN TO TEMP FILE
005362  016705  175152                          MOV     TMPCNT+2,R5     ;GET HIGH-ORDER VALUE
005366  032767  040000  175060                  BIT     #RT$CR,OTYPE    ;ARE WE TO ADD CR-LF TO OUTPUT RECORDS?
005374  001010                                  BNE     8$              ;BR IF YES
005376  032767  000001  175050                  BIT     #RT$CBL,OTYPE   ;IS THIS A COBOL FILE?
005404  001416                                  BEQ     9$              ;BR IF NOT
005406  032767  002000  175040                  BIT     #RT$RB,OTYPE    ;IS THIS A COBOL BINARY FILE?
005414  001412                                  BEQ     9$              ;BR IF NOT
005416  066704  175110                  8$:     ADD     SRTCNT,R4       ;ADD # BYTES NEEDED FOR EXTRA CONTROL CHAR
005422  005505                                  ADC     R5              ;PROPOGATE CARRY
005424  066705  175104                          ADD     SRTCNT+2,R5     ;ADD HIGH-ORDER
005430  066704  175076                          ADD     SRTCNT,R4       ;ADD FOR 2ND CONTROL CHARACTER
005434  005505                                  ADC     R5
005436  066705  175072                          ADD     SRTCNT+2,R5
005442  012700  001000                  9$:     MOV     #512.,R0        ;# BYTES PER DISK BLOCK
005446  004767  002056                          CALL    DIVIDE          ;CALC # BLOCKS NEEDED
005452  062704  000001                  7$:     ADD     #1,R4           ;GET 1 EXTRA
005456  005505                                  ADC     R5              ;PROPOGATE CARRY
005460  001402                                  BEQ     10$             ;BR IF AMT NEEDED FITS IN 16 BITS
005462  012704  177777                          MOV     #-1,R4          ;ASK FOR LARGEST AMT AVAILABLE
                                        ;  Try to open the output file.
005466                                  10$:    $NEW    #FD3,#SPC1,R4
005622  103013                                  BCC     OPNTMP          ;BR IF NO ERROR
                                        ;  CANNOT OPEN OUTPUT FILE
005624  120027  000001                          CMPB    R0,#1           ;NOT ENOUGH SPACE?
005630  001004                                  BNE     2$              ;BR IF SOME OTHER ERROR
005632                                          XXX     10              ;NOT ENOUGH SPACE
005642                                  2$:     XXX     11              ;SOME OTHER OPEN ERROR
                                        ;  THE OUTPUT FILE IS OPEN -- REOPEN TEMP FILE FOR INPUT.
005652                                  OPNTMP: $OLD    #FD2,#SPC2
005776  103004                                  BCC     8$              ;BR IF NO LOOKUP ERROR
006000                                  9$:     XXX     23              ;CAN'T REOPEN TEMP FILE
006010  005767  174512                  8$:     TST     T2SIZ           ;IS TEMP FILE 2 NEEDED?
006014  001453                                  BEQ     6$              ;BR IF NOT
006016                                          $OLD    #FD4,#SPC3
006142  103716                                  BCS     9$
006144                                  6$:     $UNLOCK                 ;UNLOCK USR MODULE
                                        ;  BEGIN SECOND PHASE OF SORT.
                                        ;  (MERGE TEMP FILE SORTED STRINGS)
006146  000167  0000006                         JMP     SORTP2
```

```
                                        ;
                                        ;  TERMINATION OF PHASE 2
                                        ;
006152  005000                          P2TERM: CLR     R0              ;SAY NO ERROR
                                        ;
                                        ;  THE XXX MACRO CAUSES A JUMP TO ABORT WITH THE ERROR CODE IN R0.
                                        ;
006154  010046                          ABORT:  MOV     R0,-(SP)        ;SAVE ERROR CODE
006156                                          $UNLOCK                 ;MAKE SURE USR IS NOT LOCKED
006160  012600                                  MOV     (SP)+,R0
006162  000167  000000G                         JMP     TERM            ;REENTER SORTI
```

```
                                .IF DF RT11
                                        .SBTTL  RSTS oriented routines
                                ;
                                ;  SETNRT
                                ;
                                ;       Set the RSTS oriented non-RT11 file parameters for the
                                ;       next file operation.  These items are the PPN, protection
                                ;       code, mode, and clustersize and are stored in a 4-word
                                ;       block whose address immediately follows the JSR R5,SETNRT
                                ;       call.  These items were saved off from a previous .CSISPC
                                ;       monitor decode and will be loaded in the "one shots" of
                                ;       the RT11 runtime's Read/Write stack (which is pointed at
                                ;       by the low core RMON word).
                                ;
006166  105767  175464          SETNRT: TSTB    RSTS            ;We running under RSTS?
006172  001420                          BEQ     10$             ;No, just ignore
006174  010046                          MOV     R0,-(SP)        ;Else get some work registers
006176  010146                          MOV     R1,-(SP)
006200  012500                          MOV     (R5)+,R0                ;Point to the saved items
006202  013701  000054                  MOV     @#RMON,R1       ;Point to the R/W stack
006206  012021                          MOV     (R0)+,(R1)+     ;Load   PPN
006210  001003                          BNE     5$
006212  017561  000000  177776          MOV     @(R5),-2(R1)
006220  012021                  5$:     MOV     (R0)+,(R1)+     ;       Protection Code
006222  012021                          MOV     (R0)+,(R1)+     ;       Mode
006224  012021                          MOV     (R0)+,(R1)+     ;       Cluster Size
006226  012601                          MOV     (SP)+,R1        ;Restore the work registers
006230  012600                          MOV     (SP)+,R0
006232  000401                          BR      15$
006234  005725                  10$:    TST     (R5)+           ;Skip the save area pointer
006236  005725                  15$:    TST     (R5)+
006240  000205                          RTS     R5
                                .ENDC
```

```
                        .SBTTL  I/O ROUTINES
                ;  FLUSH:  SUBROUTINE TO OUTPUT THE CURRENT
                ;  OUTPUT BUFFER AND SWITCH TO A NEW ONE.
                ;
006242  010146         FLUSH:  MOV     R1,-(SP)        ;SAVE SOME REGISTERS
006244  010246                 MOV     R2,-(SP)
006246  010346                 MOV     R3,-(SP)
006250  016701  174322         MOV     OBUF1,R1        ;GET ADDRESS OF BUFFER
006254  016702  174332         MOV     FLBLK,R2        ;GET ADR OF NXT DISK BLK
006260  050211                 BIS     R2,(R1)         ;SET FLINK IN BUFFER
006262  016701  174174         MOV     CUROFL,R1       ;GET CURRENT FILE #
006266  016703  174312         MOV     FILFL,R3        ;GET OLD FLINK
006272  010267  174306         MOV     R2,FILFL        ;SAVE NEW FLINK
006276  005261  000620'         INC     FILNB(R1)       ;COUNT # OF BLOCKS IN FILE
006302  004767  000050         1$:     CALL    OUTWRT          ;START WRITE TO FILE
                ;  SWITCH BUFFERS
006306  016701  174266                 MOV     OBUF2,R1
006312  016767  174260  174260         MOV     OBUF1,OBUF2
006320  010167  174252                 MOV     R1,OBUF1
006324  010167  174256                 MOV     R1,OBFEND       ;SET ADDR OF END OF BUFFER
006330  066767  174234  174250         ADD     OBUFSZ,OBFEND
                ;  INITIALIZE NEW BUFFER
006336  005021                 CLR     (R1)+           ;CLEAR FLINK
006340  005021                 CLR     (R1)+           ;CLEAR # OF RECS IN BUFFER
006342  010167  174242         MOV     R1,ORPOS        ;POS WHERE NXT REC GOES
006346  012603                 MOV     (SP)+,R3        ;RESTORE REGISTERS
006350  012602                 MOV     (SP)+,R2
006352  012601                 MOV     (SP)+,R1
006354  000207                 RETURN
                ;
                ;
                ;  OUTWRT:  ROUTINE TO WRITE THE CURRENT OUT BUFFER
                ;  TO THE DISK FILE.  WHEN CALLED R3 MUST CONTAIN
                ;  THE DISK GRANULE # WHERE THE BUFFER IS TO BE
                ;  WRITTEN.
                ;  THE BUFFER IS WRITTEN FROM THE POSITION POINTED
                ;  TO BY OBUF1.
                ;
006356  010346         OUTWRT: MOV     R3,-(SP)
006360  010446                 MOV     R4,-(SP)
006362  004767  000310         CALL    DSKCVT          ;CONVERT CLUSTER # TO FILE&BLOCK #
006366                         $WAIT   R4              ;WAIT FOR PREVIOUS WRITE TO FINISH
006500  103004                 BCC     1$              ;BR IF NO WRITE ERROR
006502                 9$:     XXX     15              ;TEMP FILE I/O ERROR
006512                 1$:     $WRITE  R4,OBUF1,OWCNT,R3
006666  103705                 BCS     9$
006670  012604                 MOV     (SP)+,R4
006672  012603                 MOV     (SP)+,R3
006674  000207                 RETURN
```

```
                                        ;
                                        ;-------------------------------------------------------------
                                        ;  DSKCVT IS CALLED TO CONVERT A TEMP FILE CLUSTER NUMBER
                                        ;  TO A BLOCK AND FILE NUMBER.
                                        ;  WHEN CALLED, R3 MUST CONTAIN THE CLUSTER NUMBER.
                                        ;  ON RETURN, R3 CONTAINS THE FILE BLOCK # AND R4 CONTAINS
                                        ;  THE CHANNEL NUMBER OF THE TEMP FILE.
                                        ;  ALL OTHER REGS ARE PRESERVED.
                                        ;
006676  010346                  DSKCVT: MOV     R3,-(SP)        ;PUSH CLUSTER #
006700  016704  173710                  MOV     BLKFC,R4        ;GET # BLOCKS/CLUSTER
006704  000401                          BR      1$
006706  061603                  2$:     ADD     (SP),R3         ;CONVERT CLUSTER # TO BLOCK #
006710  005304                  1$:     DEC     R4
006712  001375                          BNE     2$
006714  005726                          TST     (SP)+           ;CLEAN OFF STACK
006716  012704  004274'                  MOV     #FD2,R4         ;ASSUME TEMP FILE #1
006722  020367  173576                  CMP     R3,T1SIZ        ;IS THIS BLOCK IN FILE 1?
006726  101411                          BLOS    3$              ;BR IF YES
006730  166703  173570                  SUB     T1SIZ,R3        ;CALC BLOCK # IN FILE 2
006734  066703  173654                  ADD     BLKFC,R3        ;T1 MAY NOT BE MULTIPLE OF BLOCKING
006740  020367  173562                  CMP     R3,T2SIZ        ;WILL IT FIT IN FILE 2?
006744  101005                          BHI     9$              ;BR IF TEMP FILE OVERFLOW
006746  012704  004314'                  MOV     #FD4,R4         ;SET FOR TEMP FILE 2
006752  166703  173636          3$:     SUB     BLKFC,R3        ;1ST CLUSTER IS CALLED # 1
006756  000207                          RETURN
006760                          9$:     XXX     12              ;OVERFLOW OF TEMP FILE
```

```
                        .SBTTL  KEY COMPARE ROUTINES
                ;-----------------------------------------------------------------------
                ;
                ;  The following routines are called to compare two key fields of various types.
                ;  On entry, the following registers are set up:
                ;    R1 = Address of start of key field 1.
                ;    R2 = Address of start of key field 2.
                ;    R3 = Number of bytes in the key field.
                ;
                ;  On return the condition code must be left set for BGT/BLT/BEQ test.
                ;  R0, R1, R2 and R3 may be destroyed.
                ;  R4 and R5 must be preserved.
                ;
                ;-----------------------------------------------------------------------
                ;
                ;  Signed computational (binary) values stored with least significant word first.
                ;
006770  060301  KYCOMP:  ADD     R3,R1           ;POINT PAST MOST SIGNIFICANT WORD
006772  060302          ADD     R3,R2
006774  006203          ASR     R3              ;GET # WORDS TO COMPARE
006776  024142          CMP     -(R1),-(R2)     ;COMPARE MOST SIGNIFICANT WORDS (SIGNED)
007000  001404          BEQ     2$              ;BR IF MOST SIGNIFICANT WORDS ARE EQUAL
007002  000207          RETURN                  ;MOST SIGNIFICANT WORDS ARE UNEQUAL
007004  024142  5$:     CMP     -(R1),-(R2)     ;COMPARE LOW ORDER WORDS
007006  103473          BLO     SETLT           ;DO THIS AS UNSIGNED COMPARISON
007010  101067          BHI     SETGT
007012  005303  2$:     DEC     R3              ;MORE TO COMPARE?
007014  001373          BNE     5$              ;BR IF YES
007016  000207          RETURN                  ;VALUES ARE EQUAL

                ;-----------------------------------------------------------------------
                ;  Compare two binary items stored with most significant word first.
                ;
007020  006203  KYBIN:  ASR     R3              ;GET # WORDS TO COMPARE
007022  022122          CMP     (R1)+,(R2)+     ;COMPARE MOST SIGNIFICANT WORDS (SIGNED)
007024  001404          BEQ     2$              ;BR IF THEY MATCH
007026  000207          RETURN                  ;MOST SIGNIFICANT WORDS ARE UNEQUAL
007030  022122  5$:     CMP     (R1)+,(R2)+     ;COMPARE LOW ORDER WORDS
007032  103461          BLO     SETLT           ;BR IF 1<2
007034  101055          BHI     SETGT           ;BR IF 1>2
007036  005303  2$:     DEC     R3              ;MORE TO COMPARE?
007040  001373          BNE     5$              ;BR IF YES
007042  000207          RETURN                  ;VALUES ARE EQUAL
```

```
                                ;-------------------------------------------------------------------------------
                                ;  Compare two items with leading separate signs.
                                ;
007044  122127  000053  KYLSS:  CMPB    (R1)+,#'+       ;ITEM 1 POSITIVE?
007050  001407                  BEQ     1$              ;BR IF YES
007052  122227  000053          CMPB    (R2)+,#'+       ;ITEM 2 POSITIVE?
007056  001447                  BEQ     SETLT           ;1ST<0, 2ND>=0
                                ;  Both values are negative so reverse order of comparison.
007060  010100                  MOV     R1,R0
007062  010201                  MOV     R2,R1
007064  010002                  MOV     R0,R2
007066  000403                  BR      2$
                                ;  First value is positive.
007070  122227  000053  1$:     CMPB    (R2)+,#'+       ;2ND VALUE POSITIVE?
007074  001035                  BNE     SETGT           ;1ST>0, 2ND<0
                                ;  Values have same sign so compare magnitudes.
007076  005303          2$:     DEC     R3              ;COUNT DOWN SIGN CHARACTER
007100  122122          3$:     CMPB    (R1)+,(R2)+     ;COMPARE CHARACTER STRINGS
007102  001002                  BNE     9$              ;BR IF UNEQUAL
007104  005303                  DEC     R3              ;MORE TO COMPARE?
007106  003374                  BGT     3$              ;BR IF YES
007110  000207          9$:     RETURN                  ;RETURN WITH CC SET

                                ;-------------------------------------------------------------------------------
                                ;  Compare two values with trailing separate signs.
                                ;
007112  005303          KYTSS:  DEC     R3              ;DON'T COUNT SIGN CHAR IN COMPARISON LOOP
007114  060301                  ADD     R3,R1           ;POINT TO SIGN CHARACTER
007116  060302                  ADD     R3,R2
007120  121127  000053          CMPB    (R1),#'+        ;1ST VALUE POSITIVE?
007124  001407                  BEQ     1$              ;BR IF YES
007126  121227  000053          CMPB    (R2),#'+        ;2ND VALUE POSITIVE?
007132  001421                  BEQ     SETLT           ;1ST<0, 2ND>=0
                                ;  Both values are negative so reverse order of comparison.
007134  010100                  MOV     R1,R0
007136  010201                  MOV     R2,R1
007140  010002                  MOV     R0,R2
007142  000403                  BR      2$
                                ;  First value is positive.
007144  121227  000053  1$:     CMPB    (R2),#'+        ;2ND VALUE POSITIVE?
007150  001007                  BNE     SETGT           ;1ST VALUE >=0, 2ND<0
                                ;  Values have same sign.  Compare magnitudes.
007152  160301          2$:     SUB     R3,R1           ;POINT TO 1ST DIGIT OF VALUE
007154  160302                  SUB     R3,R2
007156  122122          3$:     CMPB    (R1)+,(R2)+     ;COMPARE DIGIT STRINGS
007160  001002                  BNE     9$              ;BR IF UNEQUAL
007162  005303                  DEC     R3              ;MORE TO COMPARE?
007164  003374                  BGT     3$              ;BR IF YES
007166  000207          9$:     RETURN                  ;RETURN WITH CC SET
                                ;
                                ;  Set condition to cause BGT branch to be taken.
                                ;
007170  005727  000001  SETGT:  TST     #1              ;SET CC FOR BGT
007174  000207                  RETURN
                                ;
                                ;  Set condition code to cause BLT branch to be taken.
                                ;
```

```
     58 007176  005727  177777          SETLT:  TST     #-1             ;SET CC FOR BLT
     59 007202  000207                          RETURN
```

```
                                ;-------------------------------------------------------------------------------
                                ;  Compare two values that have trailing non-separate signs.
                                ;
007204  005303                  KYDIS:  DEC     R3
007206  060301                          ADD     R3,R1           ;POINT TO SIGN CHARACTERS
007210  060302                          ADD     R3,R2
007212  111100                          MOVB    (R1),R0         ;GET SIGN CHAR FROM 1ST VALUE
007214  116000  003354'                 MOVB    SINCHR(R0),R0   ;GET UNSIGNED DIGIT
007220  002021                          BGE     1$              ;BR IF VALUE POSITIVE
007222  005400                          NEG     R0              ;MAKE CHAR POSITIVE
007224  110067  174112                  MOVB    R0,SC1          ;SIGN DIGIT FROM 1ST VALUE
                                ;  First value is negative.
007230  111200                          MOVB    (R2),R0         ;GET SIGN CHAR FROM 2ND VALUE
007232  116000  003354'                 MOVB    SINCHR(R0),R0   ;GET UNSIGNED DIGIT
007236  002357                          BGE     SETLT           ;1ST<0, 2ND>=0
007240  005400                          NEG     R0              ;MAKE CHAR POSITIVE
                                ;  Both values are negative so reverse order of comparison.
007242  116767  174074  174073        MOVB    SC1,SC2         ;REVERSE SIGN DIGITS
007250  110067  174066                  MOVB    R0,SC1
007254  010100                          MOV     R1,R0           ;REVERSE NUMBER STRING POINTERS
007256  010201                          MOV     R2,R1
007260  010002                          MOV     R0,R2
007262  000410                          BR      2$
                                ;  First value is positive.
007264  110067  174052          1$:     MOVB    R0,SC1          ;SAVE SIGN DIGIT FROM 1ST VALUE
007270  111200                          MOVB    (R2),R0         ;GET SIGN CHAR FROM 2ND VALUE
007272  116000  003354'                 MOVB    SINCHR(R0),R0   ;GET UNSIGNED DIGIT
007276  002734                          BLT     SETGT           ;BR IF 1ST>=0 AND 2ND<0
007300  110067  174037                  MOVB    R0,SC2          ;SAVE MAGNITUDE OF SIGN CHAR 2
                                ;  Values have same sign so compare magnitudes.
007304  005703                  2$:     TST     R3              ;ANY DIGITS OTHER THAN SIGN DIGITS?
007306  001406                          BEQ     4$              ;BR IF NOT
007310  160301                          SUB     R3,R1           ;POINT TO FRONT OF DIGIT STRINGS
007312  160302                          SUB     R3,R2
007314  122122                  3$:     CMPB    (R1)+,(R2)+     ;COMPARE DIGITS UP TO SIGN CHAR
007316  001005                          BNE     9$              ;BR IF UNEQUAL
007320  005303                          DEC     R3
007322  003374                          BGT     3$
                                ;  Compare sign digits.
007324  126767  174012  174011  4$:     CMPB    SC1,SC2         ;COMPARE LAST DIGITS OF VALUES
007332  000207                  9$:     RETURN                  ;RETURN WITH CC SET
```

```
                                ;-------------------------------------------------------------------------------
                                ;  Compare two dates in the form (mmddyy).
                                ;
007334  126162  000004  000004  KYDATE: CMPB    4(R1),4(R2)       ;COMPARE 1ST DIGIT OF YEAR
007342  001012                        BNE     2$                ;BR IF UNEQUAL
007344  126162  000005  000005        CMPB    5(R1),5(R2)       ;COMPARE 2ND DIGIT OF YEAR
007352  001006                        BNE     2$                ;BR IF UNEQUAL
                                ;  Years are qual.  Compare month and day.
007354  012703  000004              MOV     #4,R3             ;COMPARE 4 DIGITS
007360  122122                1$:     CMPB    (R1)+,(R2)+       ;COMPARE MONTH & DAY VALUES
007362  001002                        BNE     2$                ;BR IF UNEQUAL
007364  005303                        DEC     R3
007366  003374                        BGT     1$
007370  000207                2$:     RETURN                    ;RETURN WITH CC SET
                                ;
                                ;-------------------------------------------------------------------------------
                                ;  ROUTINE TO COMPARE TWO ASCII STRINGS WHERE LOWER CASE CHARACTERS
                                ;  ARE TO BE CONVERTED TO UPPER CASE.
                                ;
                                ;  ON ENTRY,
                                ;   R1 = ADDRESS OF STRING 1.
                                ;   R2 = ADDRESS OF STRING 2.
                                ;   R3 = NUMBER OF CHARACTERS IN EACH STRING.
                                ;  R0,R1,R2,R3 AND R5 ARE DESTROYED.
                                ;
007372  010367  173236          KYUCA:  MOV     R3,CHRCNT         ;SAVE CHARACTER COUNT
007376  112100                3$:     MOVB    (R1)+,R0          ;GET A CHAR FROM STRING 1
007400  112203                        MOVB    (R2)+,R3          ;GET A CHAR FROM STRING 2
007402  020027  000141                CMP     R0,#141           ;IS STRING 1 CHAR LOWER CASE?
007406  103405                        BLO     1$                ;BR IF NOT
007410  020027  000172                CMP     R0,#172           ;LC(Z)
007414  101002                        BHI     1$                ;BR IF NOT LC
007416  162700  000040                SUB     #40,R0            ;CONVERT CHARACTER TO UPPER CASE
007422  020327  000141        1$:     CMP     R3,#141           ;IS STRING 2 CHAR LOWER CASE?
007426  103405                        BLO     2$                ;BR IF NOT LC
007430  020327  000172                CMP     R3,#172           ;LC(Z)
007434  101002                        BHI     2$                ;BR IF NOT LC
007436  162703  000040                SUB     #40,R3            ;CONVERT CHAR TO UPPER CASE
007442  120003                2$:     CMPB    R0,R3             ;DO THE COMPARISON
007444  001003                        BNE     9$                ;BR IF FOUND DIFFERENCE
007446  005367  173162                DEC     CHRCNT            ;ANY MORE CHARS TO COMPARE?
007452  001351                        BNE     3$                ;BR IF YES
007454  000207                9$:     RETURN
```

```
                        .SBTTL  MULTIPLY & DIVIDE
                ;-----------------------------------------------------------------------
                ;  Routine to multiply the 32-bit value in R4 (low-order) and R5 (high-order)
                ;  by the 16-bit value in R0.
                ;  On return the 32-bit product is in R4-R5.
                ;  R0 is preserved.
                ;
007456  010046  MULTPY: MOV     R0,-(SP)
007460  010246         MOV     R2,-(SP)
007462  010346         MOV     R3,-(SP)
007464  010402         MOV     R4,R2           ;MOVE VALUE FROM R4-R5 TO R2-R3
007466  010503         MOV     R5,R3
007470  005004         CLR     R4
007472  005005         CLR     R5
007474  000241  1$:    CLC
007476  006000         ROR     R0              ;NEED TO ADD THIS TIME?
007500  103003         BCC     2$              ;BR IF NOT
007502  060204         ADD     R2,R4           ;ADD LOW-ORDER VALUE
007504  005505         ADC     R5              ;PROPOGATE CARRY
007506  060305         ADD     R3,R5           ;ADD HIGH-ORDER VALUE
007510  006302  2$:    ASL     R2              ;ROTATE MULTIPLYING VALUE
007512  006103         ROL     R3
007514  005700         TST     R0              ;NEED TO DO MORE?
007516  001366         BNE     1$              ;BR IF YES
007520  012603         MOV     (SP)+,R3
007522  012602         MOV     (SP)+,R2
007524  012600         MOV     (SP)+,R0
007526  000207         RETURN
```

```
                                ;-------------------------------------------------------------------------------
                                ;  Routine to divide the 32-bit value in R4 (low-order) and R5 (high-order)
                                ;  by the 16-bit quantity in R0.
                                ;  The 32-bit quotient is returned in R4-R5 and the remainder is
                                ;  returned in R0.
                                ;
007530  010146          DIVIDE: MOV     R1,-(SP)
007532  010346                  MOV     R3,-(SP)
007534  005001                  CLR     R1                      ;INITIALIZE REMAINDER
007536  012703  000037          MOV     #31.,R3                 ;SET SHIFT COUNT
007542  006304          1$:     ASL     R4                      ;SHIFT BIT OUT OF LOW-ORDER
007544  006105                  ROL     R5                      ;SHIFT THROUGH HIGH-ORDER
007546  006101                  ROL     R1                      ;AND INTO R1
007550  020100                  CMP     R1,R0                   ;GOT ENOUGH TO SUBTRACT YET?
007552  103402                  BLO     2$                      ;BR IF NOT
007554  160001                  SUB     R0,R1                   ;SUBTRACT DIVISOR
007556  005204                  INC     R4                      ;INCREMENT QUOTIENT
007560  005303          2$:     DEC     R3                      ;COUNT # BITS SHIFTED
007562  100367                  BPL     1$                      ;BR IF MORE TO DO
007564  010100                  MOV     R1,R0                   ;GET REMAINDER TO R0
007566  012603                  MOV     (SP)+,R3
007570  012601                  MOV     (SP)+,R1
007572  000207                  RETURN
                                ;
007574                  PATCH0: .BLKW   20.                     ;PATCH AREA.
        007644'         POTOP   =       .
        004510'                 .END    START
```

Symbol table

```
ABORT   006154RG    002 FILNB   000620RG    002 LOWMEM  002512RG    002 RCNT2   002510RG    002 SORTP2= ****** G
ANDFLG  003340RG    002 FILPPN  003626RG    002 MAPEND  002624RG    002 RECLEN  002500RG    002 SPCFLG  003344RG    002
AREA    002542R     002 FILRN   001440RG    002 MODE    003653RG    002 REC1    002572RG    002 SPCSIZ= 000005
BELL  = 000007          FILST   000000RG    002 MSGNAM  002470RG    002 REC2    002574RG    002 SPC1    004324RG    002
BIGEST= 177777          FLBLK   002612RG    002 MULTPY  007456RG    002 RMON  = 000054          SPC2    004336RG    002
BLKFC   002614RG    002 FLDMAP  002622RG    002 MXFILS= 000310 G        RSPCS   003662RG    002 SPC3    004350RG    002
BUFSIZ= 001000          FLUSH   006242RG    002 MXFIL2= 000620 G        RSTEXT  004222RG    002 SPC4    004362RG    002
CHNBIT= 000400          FSTIME  003652RG    002 MXKEYS= 000017          RSTS    003656RG    002 SPC5    004374RG    002
CHNEXT  004232RG    002 F$BIN = 000001 G        MXREC1  002520RG    002 RSTSCH  004242RG    002 SPC6    004406RG    002
CHNO  = 000000          F$CR  = 000004 G        MXREC2  002522RG    002 RSTSJB= 000404          SPC7    004420RG    002
CHNPPN  003624RG    002 F$INPL= 000020 G        NOERR = 000001          RT$CBL= 000001 G        SPC8    004432RG    002
CHRCNT  002634R     002 F$PAUS= 000010 G        NUMBUF  003345RG    002 RT$CR = 040000 G        SPC9    004444RG    002
CINNRT  003642RG    002 F$VLEN= 000002 G        NUMCOL  002456RG    002 RT$DBL= 000002 G        SRTCNT  002532RG    002
CINSPC  003630RG    002 GTLN    003657RG    002 NUMKY2  002476RG    002 RT$FTN= 000004 G        SRTINI= ****** G
CISRTI  004502RG    002 HB$ARS= 000006          NUMLEN  002460RG    002 RT$F11= 000010 G        STACK   003336RG    002
CISRT1  004504RG    002 HB$DFS= 000002          NXTBLK  002602RG    002 RT$OR = 000200 G        START   004510R     002
CISRT2  004506RG    002 HB$DIP= 000030          OBFEND  002606RG    002 RT$OS = 000100 G        START1  004514RG    002
COMPAR  002466RG    002 HB$DUP= 000001          OBUFSZ  002570RG    002 RT$OX = 000400 G        TERM  = ****** G
CR    = 000015          HB$RIU= 000032          OBUF1   002576RG    002 RT$RA = 001000 G        TMPCNT  002536RG    002
CSPCOF= 000376          HB$RSZ= 000004          OBUF2   002600RG    002 RT$RB = 002000 G        TONOF   005260R     002
CTRLZ = 000032          HD$DEV= 000000          OPNTMP  005652R     002 RT$TXT= 000020 G        TOPMEM  002464RG    002
CUROFL  002462RG    002 HD$$SZ= 000004          ORFLG   003341RG    002 RT$VLN= 100000 G        TSXFL   003654RG    002
CURSAV  002636RG    002 IFPNT   002616RG    002 ORPOS   002610RG    002 RT$X  = 000040 G        T1SIZ   002524RG    002
DBLKFC= 000005          INFLEN  002530RG    002 OTYPE   002454RG    002 RT11  = 000000          T2SIZ   002526RG    002
DEFEXT  004116RG    002 INRLEN  002630RG    002 OUTWRT  006356RG    002 RWDLEN  002504RG    002 UEOF    003337RG    002
DIVIDE  007530RG    002 ISMARS  002502RG    002 OVLBIT= 001000 G        SAVBUF  004126RG    002 USERRB= 000053
DSKCVT  006676RG    002 ITYPE   002452RG    002 OWCNT   002566RG    002 SAVSPC  002642RG    002 USRLOC= 000046
ENDSAV  002640RG    002 JSW   = 000044 G        PATCHO  007574R     002 SC1     003342R     002 VERSNO= 000276
EOFBUF  002620RG    002 KAD     002316RG    002 PAUSMS  003355R     002 SC2     003343R     002 V3FLAG  003655RG    002
EOFCHR  003336RG    002 KLEN    002412RG    002 PFSPC = 000002          SELADR  002626RG    002 WRDS  = 000004
EOFCOL  002632RG    002 KSTRT   002354RG    002 POBASE= 000000RG    002 SETGT   007170R     002 XBUF    004000RG    002
ERR   = 000006          KTYPE   002260RG    002 POTOP = 007644RG    002 SETLT   007176R     002 XEMT    004250RG    002
ERRLOC= 000052          KYBIN   007020RG    002 P1TERM  004520RG    002 SETNRT  006166RG    002 XXX10 = 005632R     002
FDBSIZ= 000004          KYCOMP  006770RG    002 P1TOP = ****** G        SEVERR= 000010          XXX11 = 005642R     002
FD1     004264RG    002 KYDATE  007334RG    002 P2TERM  006152RG    002 SFLAGS  002450RG    002 XXX12 = 006760R     002
FD2     004274RG    002 KYDIS   007204RG    002 P2TOP = ****** G        SINCHR= 003354R     002 XXX15 = 006502R     002
FD3     004304RG    002 KYLSS   007044RG    002 QBLK    003552RG    002 SKPCNT  002514RG    002 XXX23 = 006000R     002
FD4     004314RG    002 KYTSS   007112RG    002 QLEN  = 000003 G        SKPVAL  002516RG    002 ...V1 = 000003
FILES   004456RG    002 KYUCA   007372RG    002 RCNT1   002506RG    002 SORTP1= ****** G        ...V2 = 000061
FILFL   002604RG    002 LF    = 000012

. ABS.  000044  000   (RW,I,GBL,ABS,OVR)
        000000  001   (RW,I,LCL,REL,CON)
SORTO   007644  002   (RW,I,GBL,REL,OVR)
Errors detected:  0

*** Assembler statistics


Work  file  reads:  0
Work  file writes:  0
Size of work file:  15944 Words  ( 63 Pages)
Size of core pool:  18432 Words  ( 72 Pages)
Operating  system:  RT-11

Elapsed time:  00:00:23.48
,DB4:CBSRTO,DK:CBSRTH.CRF=DK:CBSRTH,CBSRTO/C
```

Cross reference table (CREF V05.03)

```
...V1    3-17    3-17    3-21    3-21    3-28    3-28    3-28    3-35    3-46    3-46    3-50    3-50    3-50    3-96
         3-96    3-96    3-104   3-104   3-104   3-109   3-109   3-109   5-42    5-42    5-45    5-45
...V2    3-17    3-17#   3-21    3-21#   3-28    3-28    3-28#   3-28#   3-46    3-46    3-46#   3-46#   3-50    3-50
         3-50#   3-50#   3-96    3-96    3-96#   3-96#   3-104   3-104   3-104#  3-104#  3-109   3-109   3-109#  3-109#
         5-42    5-42#   5-45    5-45    5-45#   5-45#
ABORT    2-58    3-101   3-102   3-106   3-122#  5-44    6-27
ANDFLG   2-63    2-194#
AREA     2-150#
BELL     1-40#   2-80#   2-201
BIGEST   1-56#
BLKFC    2-45    2-172#  6-11    6-21    6-25
BUFSIZ   1-34#
CHNBIT   1-53#
CHNEXT   2-234#
CHNO     1-67#   3-17    3-17    3-21    3-21    3-28    3-28    3-46    3-46    3-50    3-50    3-96    3-96    3-104
         3-104   3-109   3-109   5-42    5-42    5-45    5-45
CHNPPN   2-67    2-234#
CHRCNT   2-130#  10-26*  10-41*
CINNRT   2-234#
CINSPC   2-234#
CISRT1   2-54    2-276#
CISRT2   2-54    2-280#
CISRTI   2-54    2-272#
COMPAR   2-58    2-142#
CR       1-39#
CSPCOF   1-65#
CTRLZ    1-41#
CUROFL   2-56    2-140#  5-11
CURSAV   2-62    2-181#
DBLKFC   1-35#
DEFEXT   2-234#
DIVIDE   2-53    3-71    3-90    12-7#
DSKCVT   2-61    3-23    5-41    6-10#
ENDSAV   2-52    2-182#
EOFBUF   2-44    2-174#
EOFCHR   2-43    2-192#
EOFCOL   2-43    2-179#
ERR      1-90#   3-46*   3-96*   3-104*  3-109*  5-42    5-42*   5-42*   5-42*   5-45*   5-45*   5-45*
ERRLOC   1-49#   3-46    3-96    3-104   3-109   5-42    5-45
F$BIN    2-65    2-94#
F$CR     2-65    2-96#
F$INPL   2-49    2-98#   3-43
F$PAUS   2-65    2-97#   3-32
F$VLEN   2-65    2-95#
FD1      2-240#
FD2      2-241#  3-17    3-104   6-17
FD3      2-242#  3-46    3-50    3-96
FD4      2-243#  3-21    3-24    3-28    3-109   6-24
FDBSIZ   2-240   2-240#  2-241   2-241#  2-242   2-242#  2-243   2-243#
FILES    2-256#
FILFL    2-58    2-168#  5-12    5-13*
FILNB    2-59    2-119#  5-14*
FILPPN   2-67    2-234#  3-28    3-28    3-50    3-50    3-96    3-96    3-104   3-104   3-109   3-109
FILRN    2-59    2-120#
FILST    2-59    2-118#
FLBLK    2-48    2-171#  5-9
```

Cross reference table (CREF V05.03)

```
FLDMAP    2-64      2-175#
FLUSH     2-57      5-5#
FSTIME    2-234#
GTLN      2-234#
HB$ARS    1-74#
HB$DFS    1-72#
HB$DIP    1-75#
HB$DUP    1-71#
HB$RIU    1-76#
HB$RSZ    1-73#
HD$$SZ    1-81#
HD$DEV    1-80#
IFPNT     2-56      2-173#
INFLEN    2-44      2-157#
INRLEN    2-64      2-178#
ISMARS    2-41      2-146#
ITYPE     2-49      2-132#
JSW       1-51#     2-62
KAD       2-47      2-125#
KLEN      2-47      2-127#
KSTRT     2-47      2-126#
KTYPE     2-46      2-124#
KYBIN     2-66      7-34#
KYCOMP    2-64      7-18#
KYDATE    2-66     10-4#
KYDIS     2-60      9-4#
KYLSS     2-66      8-4#
KYTSS     2-66      8-27#
KYUCA     2-44     10-26#
LF        1-39#     2-79#     3-37
LOWMEM    2-56      2-150#
MAPEND    2-64      2-176#
MODE      2-40      2-234#
MSGNAM    2-59      2-143#
MULTPY    2-53      3-69     11-8#
MXFIL2    2-58      2-77#
MXFILS    2-57      2-76#     2-77      2-118     2-119     2-120
MXKEYS    2-78#     2-124     2-125     2-126     2-127
MXREC1    2-42      2-153#
MXREC2    2-42      2-154#
NOERR     1-56#
NUMBUF    2-55      2-199#
NUMCOL    2-55      2-134#
NUMKY2    2-41      2-144#
NUMLEN    2-55      2-135#
NXTBLK    2-48      2-167#    3-22
OBFEND    2-48      2-169#    5-20*     5-21*
OBUF1     2-47      2-165#    5-8       5-18      5-19*     5-45
OBUF2     2-47      2-166#    5-17      5-18*
OBUFSZ    2-45      2-162#    5-21
OPNTMP    3-47      3-97      3-104#
ORFLG     2-63      2-195#
ORPOS     2-48      2-170#    5-25*
OTYPE     2-49      2-133#    3-55      3-62      3-64      3-66      3-77      3-79      3-81
OUTWRT    2-57      5-15      5-39#
OVLBIT    1-52#     2-62
```

Cross reference table (CREF V05.03)

```
OWCNT     2-45      2-161#    5-45      5-45
POBASE    2-19#     2-62
POTOP     2-62      12-26#
P1TERM    2-45      3-16#
P1TOP     2-72
P2TERM    2-45      3-118#
P2TOP     2-72
PATCHO    12-35#
PAUSMS    2-201#    3-35
PFSPC     1-68#     3-17*     3-21*     3-46*     3-96*     3-104*    3-109*
QBLK      2-40      2-234#
QLEN      1-46#     2-40      2-234     2-234#
RCNT1     2-57      2-148#
RCNT2     2-57      2-149#
REC1      2-60      2-163#
REC2      2-60      2-164#
RECLEN    2-41      2-145#    3-61
RMON      1-57#     4-20
RSPCS     2-234#
RSTEXT    2-234#
RSTS      2-67      2-234#    4-15
RSTSCH    2-234#
RSTSJB    1-64#
RT$CBL    2-50      2-102#    3-64      3-79
RT$CR     2-53      2-113#    3-62      3-77
RT$DBL    2-50      2-103#
RT$F11    2-51      2-105#
RT$FTN    2-50      2-104#
RT$OR     2-51      2-109#
RT$OS     2-51      2-108#
RT$OX     2-51      2-110#
RT$RA     2-52      2-111#
RT$RB     2-52      2-112#    3-66      3-81
RT$TXT    2-50      2-106#
RT$VLN    2-53      2-114#    3-55
RT$X      2-50      2-107#
RT11      1-47#     2-82      4-1
RWDLEN    2-56      2-147#
SAVBUF    2-234#
SAVSPC    2-62      2-183#
SC1       2-196#    9-11*     9-18      9-19*     9-25*     9-40
SC2       2-197#    9-18*     9-29*     9-40
SELADR    2-64      2-177#
SETGT     7-26      7-40      8-15      8-41      8-53#     9-28
SETLT     7-25      7-39      8-7       8-33      8-58#     9-15
SETNRT    2-67      3-28      3-50      3-96      3-104     3-109     4-15#
SEVERR    1-55#
SFLAGS    2-46      2-131#    3-32      3-43
SINCHR    2-210#    9-8       9-14      9-27
SKPCNT    2-42      2-151#
SKPVAL    2-52      2-152#
SORTP1    2-71      3-12
SORTP2    2-71      3-114
SPC1      2-245#    2-256     3-50      3-96
SPC2      2-246#    2-256     3-104
SPC3      2-247#    2-256     3-28      3-109
```

Cross reference table (CREF V05.03)

```
SPC4      2-248#  2-256   3-46
SPC5      2-249#  2-256
SPC6      2-250#  2-257
SPC7      2-251#  2-257
SPC8      2-252#  2-257
SPC9      2-253#  2-257
SPCFLG    2-87    2-198#
SPCSIZ    2-245   2-245#  2-246   2-246#  2-247   2-247#  2-248   2-248#  2-249   2-249#  2-250   2-250#  2-251   2-251#
          2-252   2-252#  2-253   2-253#
SRTCNT    2-53    2-158#  3-59    3-60    3-83    3-85    3-86    3-88
SRTINI    2-72    3-8
STACK     2-46    2-87    2-187#
START     3-0#   12-27
START1    2-45    3-12#
T1SIZ     2-61    2-155#  6-18    6-20
T2SIZ     2-61    2-156#  3-19    3-27*   3-107   6-22
TERM      2-71    3-125
TMPCNT    2-53    2-159#  3-75    3-76
TONOF     3-55#
TOPMEM    2-41    2-141#
TSXFL     2-40    2-234#
UEOF      2-63    2-193#
USERRB    1-50#
USRLOC    1-54#
V3FLAG    2-63    2-234#
VERSNO    1-56#
WRDS      1-89#   5-42    5-45*   5-45*
XBUF      2-234#
XEMT      2-234#  3-28    3-28    3-46    3-50    3-50    3-96    3-96    3-104   3-104   3-109   3-109   5-45    5-45
XXX10     3-101#
XXX11     3-102#
XXX12     4-27#
XXX15     5-44#
XXX23     3-106#
```

Cross reference table (CREF V05.03)

```
$ERASE    1-638#  3-28    3-50
$EXIT     1-686#
$FDB      1-145#  2-240   2-241   2-242   2-243
$GETBO    1-402#
$GETCM    1-356#
$GETTO    1-424#
$KEEP     1-448#  3-17    3-21
$LOCK     1-328#  3-16    3-39
$NEW      1-228#  3-96
$OLD      1-260#  3-104   3-109
$PRINT    1-664#  3-35
$PRT      1-676#
$PURGE    1-465#
$READ     1-492#
$REOPE    1-309#  3-46
$SAVES    1-289#
$SPC      1-159#  2-245   2-246   2-247   2-248   2-249   2-250   2-251   2-252   2-253
$SYDON    1-674#
$SYINI    1-582#
$SYSTE    1-171#  2-234
$UNLOC    1-338#  3-34    3-111   3-123
$WAIT     1-558#  5-42
$WRITE    1-525#  5-45
...CM1    3-46    3-96    3-104   3-109   5-45
...CM2    3-28    3-28    3-46    3-50    3-50    3-96    3-96    3-96    3-104   3-104   3-109   3-109   5-45    5-45
          5-45    5-45
...CM3    3-17    3-21
...CM5    3-28    3-35    3-46    3-50    3-96    3-104   3-109   5-45
...CM7    5-45
.CHAIN    1-96#   1-99#
.CLOSE    1-58#   3-17    3-21
.CSISP    1-100#
.DELET    1-97#   3-28    3-50
.DSTAT    1-98#
.ENTER    1-96#   3-96
.EXIT     1-99#
.FETCH    1-58#
.GTLIN    1-101#
.GVAL     1-100#
.HERR     1-97#   3-28    3-50
.LOCK     1-96#   3-16    3-39
.LOOKU    1-96#   3-104   3-109
.PRINT    1-97#   3-35
.PURGE    1-99#
.QSET     1-98#
.READ     1-101#
.READW    1-100#
.RELEA    1-99#
.REOPE    1-100#  3-46
.SAVES    1-100#
.SERR     1-97#   3-28    3-50
.SETTO    1-97#
.TTYIN    1-99#   3-36
.TTYOU    1-98#
.UNLOC    1-96#   3-34    3-111   3-123
.WAIT     1-101#  5-42
```

Cross reference table (CREF V05.03)

```
.WRITE   1-101#  5-45
.WRITW   1-101#
POP      1-118#  3-17   3-21   3-28   3-28   3-46   3-46   3-50   3-50   3-96   3-96   3-104  3-104  3-109
         3-109   5-42   5-45   5-45
PUSH     1-112#  3-17   3-21   3-28   3-28   3-46   3-46   3-50   3-50   3-96   3-96   3-104  3-104  3-109
         3-109   5-42   5-45   5-45
SETNRT   1-210#  3-28   3-50   3-96   3-104  3-109
XXX      2-261#  3-101  3-102  3-106  5-44   6-27
```