PART 8 SORT PROCESSOR

The SORT processor is used to arrange the records of a disc file in ascending or descending order, according to the values of keys contained in the records.

The files handled by SORT can be TDFM files or sequential files. Records in the input file may be of variable length, but the keys must be contained in the fixed part of the record.

The SORT processor can run as a batch program, a middleground program, or a subprogram called from a user program.

SORT needs 12k words of memory (4k for the root and the longest segment of the processor, and 8k words work area) and 2 disc work files.

PROCESSING

The SORT process consists of 4 phases:

- SORT commands are read, and a parameter table is initialised.
- The input file is scanned, keys are read into the work area until this area is full, an internal sort operation is executed, and the keys are written onto the work file. This phase is repeated until the whole input file has been read.
- Keys are read from the work file, merged, and written onto the other work file. This phase is repeated until the keys are in the required order.
- The sorted keys are read; by means of information contained in the key records, the corresponding input file records are accessed and written onto the output file.

The last phase is omitted for TDFM files; a key file is produced via the special TDFM exit (see section 'TDFM Exit').

INPUT/OUTPUT

Input File

The input file for SORT may be a TDFM or a sequential file, with records of fixed or variable length. The input file can be a catalogued user file, or a temporary file. Sequential files must be contained entirely on one DAD; the sub-files of a TDFM file may be stored on different DADs.

A filecode must be assigned to the input file, either before SORT is called or as a parameter in the SORT call. If the input file is consecutive (i.e. stored on consecutive granules), the filecode must be assigned to the file before SORT is called.

Output File

The output file may be written on any medium. If it is a disc, the user must provide sufficient disc space; if not enough disc space is available, the job will be aborted.

Work Files

SORT uses two work files. The work files are temporary files, which may be located on different DADs. Both work files must be long enough to contain the whole key file.

KEYS

SORT arranges records according to key values. Keys can be character keys (occupying one or more 8-bit character positions) or arithmetic keys (occupying one or more 16-bit words). The keys may be located anywhere in the record, and may overlap each other. Up to 15 keys may be specified, but the total key length must not exceed the sector length of the work files.

EXITS

In SORT three exits are available: two user exits, and a special TDFM exit. The exits are subroutine calls to user-written subprograms, which must be linked with the SORT processor.

The first user exit (USEXI1) is entered each time a record is read from the input file; it allows the user to delete that record. The second user exit (USEXI2) is entered just before a record is written onto the output file; it allows the user to format, change or delete the current record, or to insert one or more new records. The TDFM exit is entered only when SORT processes TDFM files. The exit routine is used to store key records on disc.

The dummy exit routines provided with SORT contain a direct return to the SORT processor.

SORT can be executed in three ways:

- as a background program
- as a middleground program
- as a subprogram, called from a user program.

PROCESSOR CALLS

SORT as a background program is called by the BCP command:

```
    SRT [DUMP=[{ALL | PROG | NO}][,SIZE={<n> | MAX}]
```

The parameter SIZE reserves a work area of <n> pages (or 16 pages if MAX is specified). This parameter is used only if the background program is not memory-resident.

DUMP indicates whether a dump is required after an abort or after an exit: ALL dumps the monitor and the background machine, PROG only dumps the background machine, NO indicates that no dump is required. The default is DUMP=NO.

SORT control commands are read from /EE (catalogued procedure) or /E0. The exit code is returned in register A7.

SORT as a middleground program is called by:

```
    RUN SRT
```

The SORT processor will try to get the maximum available memory area for its work area.

SORT PARAMETERS

When SORT is called as a background or middleground program, parameters must be specified as follows:

```
    SRT  INAM={<filecode> | <filename>}[,IDAD=<DAD filecode>]
         [,IUSI=<userid>],ONAM={<filecode> | <file name>}
         [,ODAD=<DAD filecode>][,OUSI=<userid>]
         [,SWK1=<DAD filecode>]
         [,SWK2=<DAD filecode>]
          ,KEY1=(<key address>,<key length>,{AC | DC},{A | L})
         [,KEY2=(<key address>,<key length>,{AC | DC},{A | L})]
         :
         [,KEYF=(<key address>,<key length>,{AC | DC},{A | L})]
         [,MREC=<n>]
```

INAM	specifies the filecode or file name of the input file. <filename> may only be specified for a catalogued file; SORT will assign filecode /D0 to that file. If <filecode> is specified, that filecode must have been assigned previously to a catalogued file or a temporary file; in this case IDAD and IUSI parameters can be omitted.
IDAD	specifies the DAD filecode of the input file. Default: the DAD filecode specified in the :JOB command.
IUSI	specifies the user identification of the library in which the input file has been catalogued. Default: the user identification specified in the :JOB command.

ONAM) ODAD) OUSI)	define the output file, as described for the input file, but with filecode /D1 instead of /D0.
SWK1	specifies the DAD filecode of work file 1; SORT assigns filecode /D6 to that file. If the parameter is omitted, it is assumed that /D6 has been assigned to work file 1 by the user.
SWK2	specifies work file 2, as described above, with filecode /D7.
KEY1	specifies the first key: <key address> is the relative position (in characters) of the key in the record; the first position in the record = 0. <key length> is the length of the key (in characters). Note: <key address> + <key length> must not exceed the record length. AC indicates ascending key values; DC indicates descending key values; A indicates an arithmetic key; L indicates a logical (character) key.
MREC	specifies the maximum record length in the input file, as an integer number of characters; default value = 4095.

SORT called as a Subprogram

SORT can be called as a subprogram from a user program by means of:

```
    CF A14,SORTPG
```

Register A8 must point to a Sort Block, which must have been prepared by the calling program. The layout of the Sort Block is:

```
                 ------------------------------------------------------
Word 0          | E |  number of keys       |    number of input files  |
                |-----------------------|---------------------------|
Word 2          |    input filecode     |    output filecode        |
                |-----------------------|---------------------------|
Word 4          |    SWK1 DAD filecode  |    SWK2 DAD filecode      |
                |---------------------------------------------------|
Word 6          |                    reserved                       |
                |---------------------------------------------------|
Word 8          |                    reserved                       |
                |---------------------------------------------------|
Word 10         |                      MREC                         |
                |---------------------------------------------------|
Word 12         |                    reserved                       |
                |---------------------------------------------------|
Word 14         |                    reserved                       |
                |---------------------------------------------------|
Word 16         |                    reserved                       |
                |---------------------------------------------------|
Word 18         |                    reserved                       |
                |---------------------------------------------------|
Word 20         |               key address of KEY1                 |
                |---------------------------------------------------|
Word 22         | AC | L |        key length of KEY1                |
                |---------------------------------------------------|
                ~                                                   ~
                |---------------------------------------------------|
Word X          |            filecode of first sub-file             |
                |---------------------------------------------------|
Word X+2        |           last sector number of the file          |
                |---------------------------------------------------|
Word X+4        |           displacement of the last sector         |
                |---------------------------------------------------|
```

Word 0	bit 0 (E) must contain '1' when SORT deals with TDFM files. 'number of keys' is the number of keys for one record (1 - 15). 'number of input files' is the number of TDFM input files; set to 1 for sequential files.
Word 2	'input filecode': the filecode of a sequential input file; set to 0 for TDFM files. 'output filecode': filecode of the output file.
Word 4	DAD filecodes for work files 1 and 2. SORT assigns filecode /D6 to work file 1, and /D7 to work file 2. If word 2 is set to 0, it is assumed that filecodes /D6 and /D7 have already been assigned to the work files.
Word 10	maximum record length of the input file, specified as a number of characters. Default value = 4095.
Word 20	relative position, in characters, of the key in the record.
Word 22	bit 0 (AC): 1 for ascending key order; 0 for descending key order. bit 1 (LC): 1 for logical (character) key; 0 for arithmetic key. <key length> specifies the length of the key, in characters.

Note: words 20 and 22 must be repeated for each key.

The following words are specified only for TDFM files:

Word X - filecode of the first sub-file.
Word X+2 - last sector number of first sub-file.
Word X+4 - displacement of last sector.
Words X, X+2 and X+4 must be specified for each TDFM input file.

USER EXITS

In SORT two user exits are available. These exits are branches to a subroutine, by means of:

```
    CF A14,USEXI1
```

for user exit 1, and:

```
    CF A14,USEXI2
```

for user exit 2.

The subroutine associated with an exit must have an entry point and a module name identical to the name used in the exit call, and must be linked to the SORT processor.

User Exit 1

User exit 1 is called each time a record has been read from the input file. Register A8 contains the ECB address used for reading the record. The exit routine must return a status in register A7:

0: the record can be processed;
1: the record must be deleted.

Changing a record in user exit 1 is not allowed.

User Exit 2

User exit 2 is called before a record is written onto the output file. Register A8 contains the ECB address for writing the record. A status value must be returned in register A7:

0: Record is validated;
1: Record must be deleted;
2: Previous record is validated and a new record is added. The address of the corresponding ECB must be specified in register A8.
3: A new record is added; instead of returning to SORT the exit routine loops to itself, so more records can be added.

TDFM Exit

When SORT handles TDFM files, key records are delivered to a TDFM exit in the final merge phase. The calling sequence for the TDFM exit is:

```
    CF A14,EDFM
```

with A8 containing the key record address in memory. Register A7 contains the address of the Sort Block (relevant only if SORT has been called as a subprogram).

The layout of a key record is:

key value	file number	sector number in input file	displacement in sector

The length of a key record depends on the key length: `key value' occupies as many characters as all keys together, rounded up to an even value, the other entries occupy one word each.

The last key record sent to the TDFM exit contains `:EOF'.

ERROR MESSAGES

Parameter Errors

```
* * * * UNKNOWN COMMAND
* * * * INPUT COMMAND I/O ERROR
* * * * INVALID DELIMITER AFTER THE COMMAND
* * * * SYNTAX ERRORS
A MINIMUM OF 8K WORDS ARE REQUIRED FOR SORT WORK AREA
KEY POSITION ERROR
KEY LENGTH ERROR
INVALID KEY PARAMETER
ASSIGN ERROR ON INPUT FILE (STATUS=XX) (see below)
ASSIGN ERROR ON OUTPUT FILE (STATUS=XX) (see below)
ILLEGAL INPUT FILE CODE
ILLEGAL OUTPUT FILE CODE
ASSIGN ERROR ON SWK1 (STATUS=XX) (see below)
ASSIGN ERROR ON SWK2 (STATUS=XX) (see below)
SORT WORK FILE IS NOT DFM
ARITHMETIC KEY LENGTH SHOULD BE EVEN
ARITHMETIC KEY POSITION SHOULD BE A WORD
ONLY 1 INPUT FILE IS ALLOWED
ILLEGAL INPUT FILE CODE /XX
```

Errors in Key Input Phase

KEY LENGTH TOO BIG
I/O ERROR ON INPUT FILE
RECORD LENGTH TOO BIG
RECORD LENGTH TOO SMALL
I/O ERROR ON SWK1

Errors in Merge Phase

I/O ERROR WHILE READING SORT WORK FILE
I/O ERROR WHILE WRITING ON SORT WORK FILE

Errors in Output Phase

I/O ERROR (SORT WORK FILE)
I/O ERROR (INPUT FILE)
I/O ERROR (OUTPUT FILE)

Status Values in Assign Errors

1	Disc I/O error
2	Dynamic Area overflow; File Code Table cannot be created
3	DAD unknown
4	Device unknown
5	Required granules not available
6	File name unknown
7	Second filecode unknown or already assigned
8	Permanent filecode must not be assigned
9	Invalid assign type
A	Userid not found in DAD catalogue
B	Illegal file type
C	Illegal filecode
D	Filecode already assigned to an attached device or file
E	File Code Table overflow
F	Too many granules requested for the sector size.

Example 1: Linking SORT Modules with User Exit Routines

Command sequence:

```
SCR FCOD=/D5
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=SOROOT
NOD SORT
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=ERMES
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=SOPAR
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=CCREAD
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=CCPRIN
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=CCPOST
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=BLANKW
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=INMOD
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=INITFG
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=CCGCS
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=R:EXAS
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=DEBI
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=CCDCOD
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=DECMES
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=CCGNXC
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=CCCHCK
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=R:ASEX
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=MUST
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=ZEROS
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=CWDS
NOD SORT
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=SOINT
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=R:EXAS
NOD SORT
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=SOINP
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=BLKREC
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=KEYREC
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=USEXI1
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=CWKEY
NOD SORT
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=SOMERG
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=CWKEY
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=EDFM
NOD SORT
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=SOFMT
INC LIBR=SRTLIB,DAD=/F0,USID=MASUP,MNAM=USEXI2
LKE
OPT MAP=YES,CATL=<name>
```

Example 2: Linking SORT for Use as a Subprogram

```
SCR FCOD=/D5
INC LIBR=SRTLIB,MNAM=SOROOT
INC LIBR=SRTLIB,MNAM=SORTPG
INC LIBR=SRTLIB,MNAM=DEBI
NOD SORT
INC LIBR=SRTLIB,MNAM=SOINT
INC LIBR=SRTLIB,MNAM=R:EXAS
NOD SORT
INC LIBR=SRTLIB,MNAM=SOINP
INC LIBR=SRTLIB,MNAM=BLKREC
INC LIBR=SRTLIB,MNAM=KEYREC
INC LIBR=SRTLIB,MNAM=DFMREC
INC LIBR=SRTLIB,MNAM=USEXI1
INC LIBR=SRTLIB,MNAM=CWKEY
NOD SORT
INC LIBR=SRTLIB,MNAM=SOMERG
INC LIBR=SRTLIB,MNAM=CWKEY
INC LIBR=SRTLIB,MNAM=EDFM
NOD SORT
INC LIBR=SRTLIB,MNAM=SOFMT
INC LIBR=SRTLIB,MNAM=ISEXI2
LKE
OPT MAP=YES,CATL=<name>
```