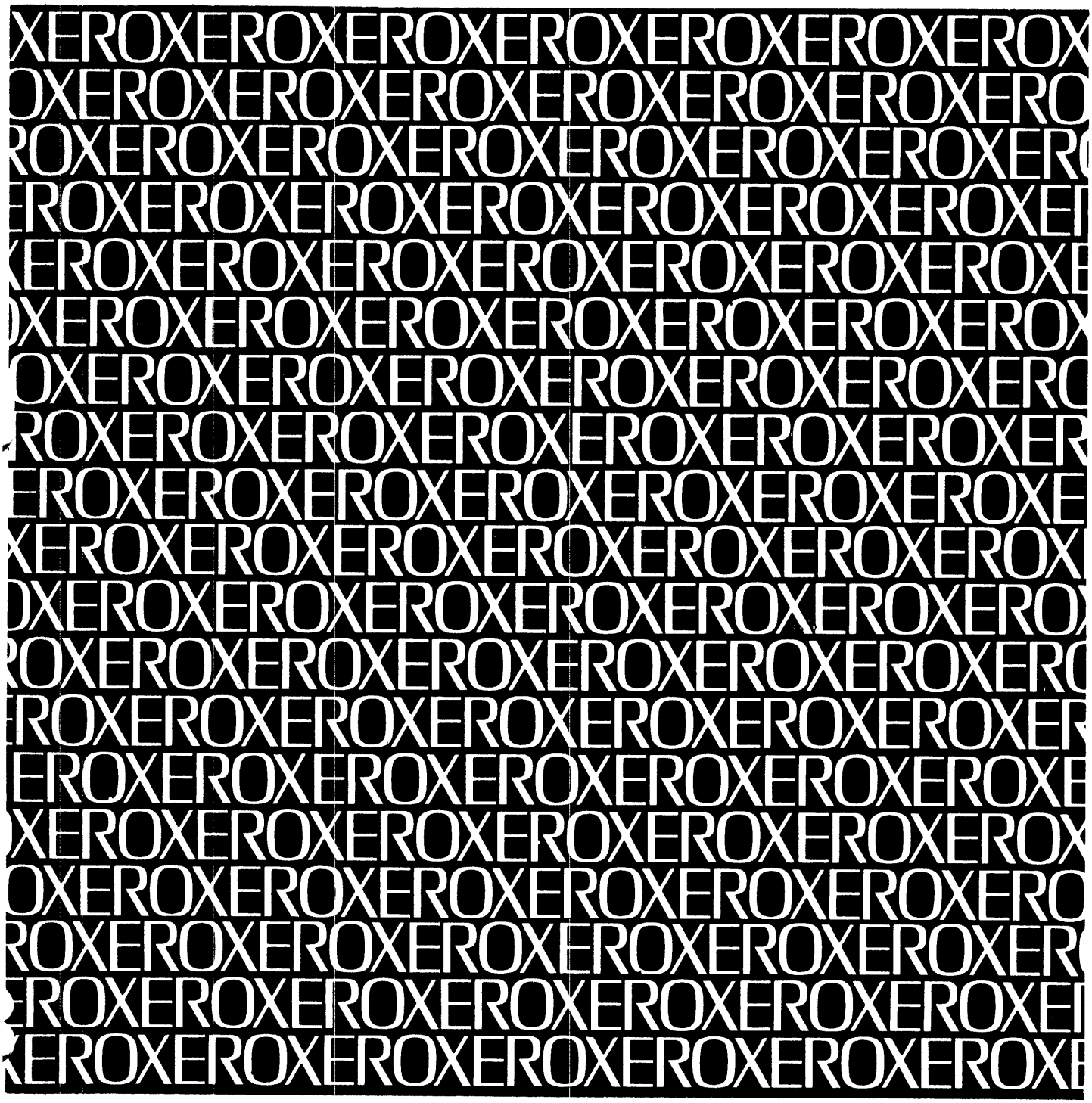
Xerox Sort and Merge (for CP-V/BPM)

Xerox 560 and Sigma 5-9 Computers

Language and Operations Reference Manual



90 11 99H

XEROX

Xerox Sort and Merge (for CP-V/BPM)

Xerox 560 and Sigma 5-9 Computers

Language and Operations

Reference Manual

99 11 99H

June 1975

File No.: 1X53

Printed in U.S.A.

REVISION

This publication is a revision of the Xerox Sort and Merge/Reference Manual for Sigma 5-9 Computers, Publication 90 11 99G (dated October 1973). The manual incorporates changes documenting the F00 version of the Xerox Sort and Merge processor for version C01 of the CP-V operating system and for version H01 of the BPM operating system. Changes are indicated in the manual by a vertical line in the margin of the page.

RELATED PUBLICATIONS

Title	Publication No.
Xerox 560 Computer/Reference Manual	90 30 76
Xerox Sigma 5 Computer/Reference Manual	90 09 59
Xerox Sigma 6 Computer/Reference Manual	90 17 13
Xerox Sigma 7 Computer/Reference Manual	90 09 50
Xerox Sigma 8 Computer/Reference Manual	90 17 49
Xerox Sigma 9 Computer/Reference Manual	90 17 33
Xerox ANS COBOL/LN Reference Manual	90 15 00
Xerox ANS COBOL/OPS Reference Manual	90 15 01
Xerox Control Program-Five (CP-V)/TS Reference Manual	90 09 07
Xerox Control Program-Five (CP-V)/OPS Reference Manual	90 16 75
Xerox Control Program-Five (CP-V)/TS User's Guide	90 16 92
Xerox Control Program-Five (CP-V)/SM Reference Manual	90 16 74
Xerox Control Program-Five (CP-V)/SP Reference Manual	90 31 13
Xerox Control Program-Five (CP-V)/BP Reference Manual	90 17 64
Xerox Control Program-Five (CP-V)/RP Reference Manual	90 30 26
Xerox Batch Processing Monitor (BPM)/BP, RT Reference Manual	90 09 54
Xerox Batch Processing Monitor (BPM) and Batch Time-Sharing Monitor (BTM)/OPS Reference Manual	90 11 98
Xerox Batch Processing Monitor (BPM) and Batch Time-Sharing Monitor (BTM)/SM Reference Manual	90 17 41

Manual Content Codes: BP – batch processing, LN – language, OPS – operations, RP – remote processing, RT – real-time, SM – system management, SP – system programming, TP – transaction processing, TS – time-sharing, UT – utilities.

CONTENTS

APPENDIXES

FIGURES

TABLES

EXAMPLES

COMMAND SYNTAX NOTATION

Notation conventions used in command specifications and examples throughout this manual are listed below.

Notation	Description
lowercase letters	Lowercase letters identify an element that must be replaced with a user-selected value. CRndd could be entered as CRA03.
CAPITAL LETTERS	Capital letters must be entered as shown for input, and will be printed as shown in output. DPndd means "enter DP followed by the values for ndd".
[ ]	An element inside brackets is optional. Several elements placed one under the other inside a pair of brackets means that the user may select any one or none of those elements. [KEYM] means the term "KEYM" may be entered.
. . .	The horizontal ellipsis indicates that a previous bracketed element may be repeated, or that elements have been omitted. name[,name]. . . means that one or more name values may be entered, with a comma inserted between each name value.
Numbers and special characters	Numbers that appear on the line (i.e., not subscripts), special symbols, and punctuation marks other than dotted lines, brackets, braces, and underlines appear as shown in output messages and must be entered as shown when input. (value) means that the proper value must be entered enclosed in parentheses; e.g., (234).

1. SORT

INTRODUCTION

A fundamental process in data manipulation is the rearrangement, or sorting, of records in a file to a predetermined order. The Sort program operates under Control Program-Five (CP-V) or the Batch Processing Monitor (BPM), on the Xerox 560 and Sigma 5-9 computers. Design of the package permits any program compiled by COBOL, FORTRAN, or Meta-Symbol to call Sort as a subroutine during normal program execution. Sort parameters supplied at run time define the characteristics of each job.

Sort is designed to be device-independent. While intermediate (scratch) Sort files default to RAD or disk, they may be assigned to from 1 to 17 disk packs or from 3 to 17 magnetic tape units via an ASSIGN control command. Sort uses a backward-read technique to eliminate the time spent waiting for rewinds when the intermediate files are assigned to sequential storage media (e.g., magnetic tapes). Sort allows the user complete flexibility in using his own hardware configuration to greater advantage. Mixing of tape, nonrandom disk (i.e., disk written sequentially), and nonrandom RAD storage for intermediate files is allowed, but the user may have to reserve one tape unit to be used for his final output file if it is to be written on magnetic tape (see Chapter 4). Sort releases the unused scratch file at the beginning of the final pass; but with scratch files assigned to RAD, disk, and tape, there is no guarantee that the unused file available during the output pass will be one assigned to magnetic tape, unless the user is careful in making his intermediate DCB assignments (see Chapter 4). When intermediate scratch files are assigned to magnetic tape, the ASSIGN control command should specify either the parameter (DEVICE,9T) or the parameter (DEVICE,7T) rather than the parameter LABEL in order to allow Sort to control intermediate file record size and blocking, thereby increasing Sort speed.

FEATURES

Major features of the Sort program are summarized as follows:

1. Files can consist of records formatted as ANSI, monitor (i.e., BPM or CP-V), or user.

2. Fixed-length records in blocked or unblocked format can be handled. (In monitor-formatted files, records are unblocked to sort.)

3. Variable-length records can be handled in blocked or unblocked ANSI format and unblocked user or monitor format.

4. Sorting can be accomplished on multiple key fields, in either ascending or descending sequence.

5. Records containing from 1 to 16 key fields can be sorted.

6. The read-backward feature of 9-track tapes is used when intermediate work files are on sequential storage media.

7. Sorting can be performed on the following types of key field data:

   a. Alphanumeric

   b. Binary (including normalized floating-point numbers)

   c. Zoned decimal

   d. Packed decimal

8. User own-code can be inserted at specified points, permitting record modification, record deletion, record insertion, and access to multiple-block header and trailer records. When records are equal on all specified keys, input order is preserved.

9. User-specified character-collating sequence can be utilized.

10. I/O on all files is buffered.

11. Intermediate work files can be on sequential storage media (i.e., magnetic tape and nonrandom disk) or on random storage media (i.e., direct-access disk and RAD), but not on a mixture of sequential and random storage media. However, any combination of sequential media may be used at one time, as may any combination of random media.

12. If for any reason Sort encounters an error condition, Sort will set the Step Code to 6. The user may therefore test this code via !STEP (CP-V only) and selectively execute the remaining job steps.

MEMORY ALLOCATION

Memory allocation is calculated in the first phase of Sort for use in subsequent phases. Results of the calculations are stored as control table entries. The amount of working storage available for the second phase of the program is developed first. The "tournament" size is then obtained from the working storage based on the file's characteristics.

PROGRAM ORGANIZATION

Sort is in reality a dual processor system. The two processors are referred to throughout this manual as the sequential Sort processor and the random Sort processor

("random" meaning direct-access). Which of these processors is used is determined by the type of storage media specified by the user for intermediate work files. If work files are exclusively on sequential storage media (i.e., magnetic tape and disk written sequentially), the sequential Sort processor is used; and if the work files are exclusively on random storage media (i.e., disk and RAD), the random Sort processor is used.

The determination to call the sequential processor or the random processor is performed by a preprocessor named SPRE. SPRE is summoned in one of two ways: when a SORT command is given or when Sort is linked to via the M:LINK procedure. (The M:LINK procedure is described in the following Xerox manuals: CP-V/BP Reference Manual, 90 17 64, and BPM/BP, RT Reference Manual, 90 09 54.) SPRE analyzes the intermediate work file DCBs. If any work file is assigned to tape or nonrandom disk storage, the sequential Sort processor is branched to; otherwise the random Sort processor is branched to.

SPRE also reads the Sort control parameter commands when it is called in the batch mode. The Sort control parameter commands are REC, BLOCK, FILE, KEYS, NOTE, LIMIT, and TRAN. These commands and their functions are discussed later in this chapter.

Both the sequential and random Sort processors that can be optionally invoked by SPRE contain four basic phases which are described in the following paragraphs. SPRE is not present when Sort is used as a co-resident processor (see the Sort structure description in Chapter 3).

PHASE I

The first phase acquires various user-supplied parameters (e.g., file characteristics, record characteristics, default options, etc.), checks them for consistency, and builds control tables for use by the following phases. This phase also determines the extent of machine resources available and sets intermediate storage blocking factors and working storage values for the following phases.

PHASE II

The second phase builds the key comparison routines and, in the sequential Sort processor, builds the tournament with the initial portion of the input file.

PHASE III

The third phase reads the remainder of the input file and forms strings of ordered records that are stored on the intermediate files. In the random Sort processor, merging also takes place in this phase, up to the final merge.

PHASE IV

The fourth phase merges the initial strings into successively longer strings until each file contains only one string. At this point the fourth phase merges these strings onto the final output file as one ordered string. In the random Sort processor, only the final merge is performed in this phase.

OPERATING MODES

Sort can operate in either of two modes: job mode or program call mode. For the job mode of operation, the user calls Sort out of the system library via a SORT processor control command (monitor loader). For the program call mode of operation, Sort is called by user programs written in ANS COBOL, FORTRAN, or Meta-Symbol. In this case the only requirement is that the calling program must be able to manipulate certain registers. This requirement may necessitate including machine language instructions in FORTRAN, for example, to set up exit and return conditions. In a COBOL program, the necessary linkage to Sort is generated at compile time when a SORT verb appears in the user source program. During execution, the sort function is accomplished with no further action required by the user.

When Sort is called as an independent processor in the batch job mode, the Sort control module is loaded and its execution is initiated. The user's specifications are then read from the control input device, and processing is initiated as explained below.

INTERNAL SORTING METHODS

In the internal sorting method employed by Sort, the input file is read and strings of ordered records are developed and output to intermediate files. Internal sorting is accomplished through use of the "replacement-selection tournament" technique. The records themselves are not moved during the tournament play, but a table of index flags is manipulated to keep track of losing "players".

FILE ORGANIZATION

Files that are to be sorted may consist of fixed-length or variable-length logical records and can fall into three major categories: ANSI-, monitor-, and user-formatted. ANSI-formatted files may be of fixed or variable length, blocked or unblocked. Monitor-formatted files may be of fixed or variable length and are unblocked to sort. User-formatted files may be of fixed length, blocked, or of variable length, unblocked. The last physical record in the

file may be shorter than a full block if the proper multiple of logical records is not present. Padding characters added to fill out the last block will be treated as data and sorted accordingly.

General characteristics of ANSI-, monitor-, and user-formatted files are outlined below.

ANSI-FORMATTED FILES

The format and structure of ANSI-formatted files conform to the American National Standard for Magnetic Tape Information Interchange, except that the decimal tape format is not processed. The Block Header File is restricted to 0-byte length for fixed-length format, and to a length of four bytes for variable-length format. If a Header 2 label (HDR2) is not present, appropriate file description information must be supplied via ASSIGN control cards (see the CP-V/BP Reference Manual, 90 17 64). Only one 80-byte user header label (UHL1) or trailer label (UTL1) is permitted per file volume. If user labels are present, the user must provide program instructions for checking input labels and creating output labels. When using ANSI-formatted files and the DROPBLOCK FEATURE, the user must put (ABCERR) on !ASSIGN cards as appropriate. The user must be sure that the DCB for F:SORTIN and F:SORTOUT has the proper BLKL and FMT values in the DCB. The .BLOCK card is not necessary for ANSI files.

MONITOR-FORMATTED FILES

The packing structure of monitor-formatted files is determined by the monitor (that is, by CP-V or BPM). The file structure cannot be changed by user specifications. The first group of records in the file contains the monitor label which contains information, such as the file name, user account number, and logical record blocking factor. If the file is on tape, an optional user label (255 bytes maximum) is also present.[†] The user label will be ignored unless the user explicitly indicates that "own-code" has been provided to process the label and to generate a new label for the sorted file.

If the input file resides on the RAD or disk pack, the user may elect to destroy the input file at the end of the input phase by specifying RELEASE in the ASSIGN command. This will allow the released RAD/disk area to be available for additional Sort resources. It should be emphasized that this option precludes rerun of the job if a Sort error or abort occurs during the final output phase.

USER-FORMATTED FILES

The packing structure of user-formatted files is explicitly controlled by the user. The Sort requires blocking factors from the user in order to provide automatic blocking and deblocking of logical records. The file may or may not be headed by a user header label. User header labels may be single-block or multiple-block headers and may be in any format the user requires. A single-block header label is defined to be the first physical block of the file and cannot exceed the data block size (or exceed 255 bytes). If a label is present, the user must provide program instructions for checking the input label and for generating the output label.

If file header labels are used, the user may make program modules available in the user library to process input and output headers. If the file is ANSI-formatted or monitor-formatted, the user header is assumed to be in the user label area. If the file is user-formatted, the first physical block on each physical volume will be treated as a header. If the user does not provide program modules to be process header labels, Sort will warn him via a message on the LL device, skip the label record, and proceed to Sort. If user header labels are multiple-block, they must be terminated by a file mark. In the absence of user-own code, Sort will do a PFILE and then proceed.

If volume trailer records are specified, the user may make program modules available in the user library to process input and output trailers. Trailers appear at the end of each physical volume. If the user does not provide program modules to process trailer labels, Sort will warn him via a message on the LL device and will ignore the trailer labels.

Included in the Sort program are various secondary references that must be linked to user own-code when headers, trailers, or access to data records is specified. The user assembles his program as independent relocatable object modules and then builds an overlay structure (with his account and new load module name) that merges his program with the Sort root segment SROOT independent of whether he uses the sequential sort or the random sort. This new load module should be called something other than SORT. This results in a unique program for that particular sorting task. An example appears in Appendix A.

The user's program must contain external definition of entry points to be used as follows:

When Sort Specification Is	Sort Branches to User's Definition of	When
Header Labels	S:INHED	After reading an input user header
	S:OUHED	Before writing an output user header
Trailer Labels	S:INTRL	After reading an input user trailer
	S:OUTRL	Before writing an output user trailer

[†] See the BPM/BP,RT Reference Manual, 90 09 54, appendix titled "File Organization"; or the CP-V/BP Reference Manual, 90 17 64, Chapter 2, as appropriate.

When Sort Specification Is	Sort Branches to User's Definition of	When
Input and Output Own-Code	S:INUSO	After accessing each input data record
	S:OUSO	Before outputting each sorted data record

User own-code modules may include their own DCBs and input/output functions. Own-code modules remain resident during the entire job, with a resulting decrease in working storage available to Sort. User definitions are explained below.

INPUT PHASE OWN-CODE

Sort cannot protect the integrity of its input buffer or record areas during the execution of user own-code instructions. The user must take all precautions necessary to prevent his program from modifying areas outside his jurisdiction.

S:INHED This external definition is assumed to exist in a user module appended to SSP (sequential) or SRP (random) whenever a user header label is present. If the user has ANSI, monitor, or single-block user header labels, the user's program will be entered after each input file header is read by Sort. The starting location of an input buffer will be passed on and the own-code will be responsible for verifying its contents. If the user has specified multiple-block user header labels, Sort will enter the user's own-code routine after performing the read operation. (It is up to the user to read his remaining labels, perform any desired validity checking, and position the file to the beginning of data information.) If needed, the user may perform his own read-in of parameters (such as a data card) against which the input header can be compared. After verification, the user will return to Sort at a specified linkage point. The user's header is truncated if its length exceeds 80 bytes (ANSI format), 255 bytes (monitor format), or the length of the data block (user format).

S:INTRL This external definition is assumed to exist in a user module appended to SSP (sequential) or SRP (random) when trailer records following each volume of the input file are present. The user's program will be entered after each input trailer is read by Sort.

At the time of entry to S:INHED and S:INTRL, the following conditions will exist:

Register 5 contains the return address (B *R5).

Register 6 contains the byte address of Sort input buffer.

Input buffer byte 1 contains the length of input header trailer.

Input buffer byte 2 contains the start of header/trailer record.

If the user specifies multiple-block user header labels for a user-formatted file, only Register 5 will be set at entry to S:INHED.

S:INUSO This external definition is assumed to exist in a user module appended to SSP (sequential) or SRP (random) when access to input data records is desired. The user's program will be entered after each input logical record has been acquired but before that record has been entered into Sort. The starting location of the logical record will be passed on, and the user may perform any of the following operations on the data:

1. Summing, rearranging, coding and decoding, etc.

2. File updating (deleting, replacing, and inserting records). The user may modify the record passed to him by Sort, but if he expands the record's length he must first move it to his own record area in order to avoid destroying records following this one in the current input stream. If the user increases record length at this time, the expanded length must have been specified as input record length. The user must not decrease record size such that any portion of a specified key will be truncated or "lost".

After completion of the user's processing of each logical record, the own-code routine should return to Sort as described below. At the time of entry to S:INUSO, the following conditions will exist:

Register 5 contains the normal return address to the Sort program to enter the record into the sorting process (B *R5). When the user is inserting or replacing records, bits 18-31 must contain the record's length as the number of bytes in the record.

Register 6 contains the byte address of the last record read. A zero indicates the file is exhausted. When Sort executes the user's own-code module after the end-of-file has been encountered on the input file (indicated by a 0 in register 6), it is only to allow the user's routine to close off its operations.

Register 7 contains the return address to Sort to be used for deletion, insertion, replacement, or record length modification. When the user returns to the address specified in R7, he must provide in R6 the record's byte address (bits 13-31) and an operation code in bits 00-07. The operation codes are

00 delete record.

01 replace record with record in location specified in R6.

02 insert record in location specified in R6 and return to user's own-code with original record address in R6.

Register 8 contains the record length in bytes.

OUTPUT PHASE OWN-CODE

Sort cannot protect the integrity of its output buffers or record areas during the execution of user own-code instructions. The user must take all precautions necessary to prevent his program from modifying areas outside his jurisdiction.

S:OUHED This external definition is assumed to exist in a user module appended to SSP (sequential) or SRP (random) when a user wishes to prepare an output header label. The user's program will be entered before any data record is written on each physical volume of the output file. The user may construct his header in a work area contained within S:OUHED and return to Sort with the byte address of that area. The first byte of the area contains the number of bytes in the header (this byte is not written to the output file), followed by the header information itself. Entry is made to S:OUHED just before the file (volume) is opened. If the user has specified multiple-block user header labels, Sort will enter the user's own-code routine without performing any WRITE operation. When the user returns to Sort, the last header label will be written.

S:OUTRL This external definition is assumed to exist in a user module appended to SSP (sequential) or SRP (random) when a user wishes to prepare trailer records. The user's program will be entered after each volume has been written. User processing follows the same rules as for header labels.

The following conditions are assumed to exist in connection with S:OUHED and S:OUTRL:

At entry, Register 5 contains the return address (B *R5).

At exit, Register 6 contains the byte address of a user's output buffer (set by user). This buffer must start on a word boundary.

Output buffer byte 1 contains the length of header or trailer.

Output buffer byte 2 contains the start of the header or trailer record.

S:OUSO This external definition is assumed to exist in a user module appended to SSP (sequential) or SRP (random) when access to output data record is desired. The user's program will be entered before each logical record is written (after it has been selected for output). The starting location of the logical record will be passed on. The user's option and interface are the same as those specified under S:INUSO, except that at end-of-file (EOF) the user may make an unlimited number of insertions.

Note: If the user modifies either content or relative location of a key field in his own-code routine and has selected the sequence check option, he will produce an out-of-sequence condition causing Sort to abort. Insertion of records may also produce this condition if the user is not careful to preserve the sequence ordering obtained by Sort. Insertion and replacement records must start on word boundaries when sequence checking is requested.

If the S:OUSO user specifies sequence check and is modifying the output record length, he must specify the maximum output record length to Sort.

KEY FIELDS

Records to be sorted may contain from 1 to 16 key fields. A key field must be aligned on byte boundaries and be entirely contained within the minimum record length specified. A key field may contain the following types of data:

Data Type	Length of Key Field
Alphanumeric	A maximum of 255 8-bit EBCDIC characters. Unsigned, zoned decimal field should be placed in this category for increased efficiency in the comparison process.
Binary	A maximum of 8 bytes.
Packed decimal†	A maximum of 31 decimal digits, plus sign, packed into 16 bytes.
Zoned decimal†	A maximum of 31 decimal digits, plus sign, in EBCDIC form contained in 31 bytes.

†The user's system must have decimal arithmetic capability to handle this type of key field.

Individual key fields may be sorted in ascending or descending sequence. Comparison results are algebraic for decimal and binary keys, and absolute EBCDIC for alphanumeric keys.

The collating sequence value of the characters used in alphanumeric, packed decimal, or zoned decimal key fields may be transformed according to a user-supplied table of sequence values before the key fields are sorted. If character sequence translation is specified as a Sort parameter, the user must supply one or more key translation control cards showing the desired sequence changes to the standard EBCDIC characters set. In all cases where records are equal on all keys, Sort preserves the order of the input file.

OPERATING IN THE JOB MODE

Sort operates in the job mode when called via the SORT processor control command. The preprocessor SPRE is then called in and determines the type of sort desired (by examining the DCBs for the intermediate work files). It also reads the Sort parameter control cards which are discussed in the following paragraphs. These cards are read from the control device.

The parameter cards may appear in any sequence in the control deck following the SORT command. All but the NOTE parameter card may be used only once in the control deck; the NOTE card may be used as many times as required.

Each parameter control card must start with the following:

1. A period in column 1.
2. The name of the parameter control card (e.g., NOTE) beginning in column 2.
3. At least one blank following the name of the card.

If a parameter requires more space than is available on a card, it can be continued on one or more following cards (permitted only with the KEYS and TRAN parameters). When a parameter is to be continued on another card, the following rules apply:

1. Each card that is to be continued on another card must be terminated with a semicolon.
2. Each continuation card must have a period in column 1 and a blank in column 2.

REC This card specifies record length and whether or not the records in the sorted output file are to be sequence checked. It has the form

```
. REC (input, output, SEQ)
```

where

input specifies the input file record length, expressed in number of bytes.

output specifies the output file record length, expressed in number of bytes, that will be used during sorting and that will be written. A variable-length file can become fixed-length with this feature. If the length of the output record is greater than the length of the input record, unpredictable results can occur.

SEQ indicates that the records in the sorted output file are to be sequence checked.

The user must always provide the input record length. If the output record length is not specified, it is assumed to be the same as the input record length. In a variable-length file, record length is the maximum physical record length.

BLOCK This card specifies blocking factors for blocked files on tapes other than ANS labeled tapes; it is not needed for unblocked files. It has the form

```
. BLOCK (input, output, DROP)
```

where

input specifies the input blocking factor; that is, the number of records per block. The number can consist of up to three digits (999).

output specifies the output blocking factor; that is, the number of records per block. The number can consist of up to three digits (999).

DROP indicates that bad blocks are to be dropped. (A bad block is a block in which an I/O error is encountered during a read operation.) This option may appear anywhere within the parentheses.

The user must always specify the input blocking factor on blocked files. If the output blocking factor is not specified, it is assumed to be the same as the input blocking factor. This card is unnecessary for ANSI files if the BLKL field for the F:SORTIN or F:SORTOUT DCBs has the proper block size.

FILE This card specifies the number of records to be sorted and the records and header labels to be skipped (that is, not processed). It has the form

```
. FILE (SKIP, number), (SORT, number), (HDR, x)
```

where

SKIP, number specifies the number of records to be skipped in the file before processing begins. The number can consist of up to six digits.

SORT, number specifies the number of records to be sorted. The number can consist of up to six digits.

HDR, x specifies the input header labels to be skipped. The x can consist of an F or any digit from 1 through 9. An F indicates to skip past the file of header labels to the data file. A digit indicates to skip past the indicated number (1-9) of header labels. If there are no header labels, omit this option from the .FILE card.

The SKIP, SORT, and HDR parameters may appear in any order. The FILE card is optional.

KEYS This card specifies the sort key characteristics. It has the form

```
. KEYS (origin, length, type, direction, TRAN)
      [, (origin, length, type, direction, TRAN)]. . .
```

where

origin specifies the starting byte position in the record. The first byte is 1, the second byte is 2, etc.

length specifies the key length. Key length is expressed in number of bytes and can consist of up to three digits.

type indicates the sort key type and can be any of the following:

AN	indicates alphanumeric
PD	indicates packed decimal
ZD	indicates zoned decimal
BN	indicates binary
BA	indicates absolute binary (signs are ignored). If this feature is used and the high-order bit of the key is on, and no other bits are on, a fixed-point overflow will cause an abort.

If type is omitted, alphanumeric type is assumed.

direction indicates the direction of the sort (A = ascending sort, D = descending sort). If direction is omitted, an ascending sort will be done.

TRAN indicates that the translation table is to be used with the key field.

NOTE This card allows the user to insert comments into the Sort parameter control deck. A NOTE card may not be continued onto other cards, but any number of NOTE cards may be used. This card has the form

```
. NOTE [any text]
```

LIMIT This card specifies the limits for the DCBs, the amount of memory to request, and intermediate device handling. It has the form

```
.LIMIT (PAGES,number), (DCBS,number), (REM)
      , (DUMP), (TBUF,number)
```

where

PAGES,number specifies the number of pages in memory the sort is to request. If this option is omitted, the sort will request all of memory. The PAGES option performs the same function as the CORE option on the !LIMIT monitor control card (see the BPM/BP,RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate). Thus the user may specify core size via the !LIMIT monitor control card of the .LIMIT sort parameter control card, or both. (If the user specifies core size on both cards, the sort parameter card specification should be within the core size specified by the monitor control card. If it is not, the core size designated by the monitor control card takes precedence.) PAGES is in pages, where CORE is in K units.

DCBS,number specifies the number of intermediate DCBs to be used. From 6 to 17 DCBs can be specified for a random sort, and from 3 to 17 DCBs can be specified for a sequential sort. If DCBs is omitted for either sort, 6 DCBs will be assumed. For a sequential sort if less than the number of DCBs specified are assigned via ASSIGN control commands, the balance will be opened on public disk storage.

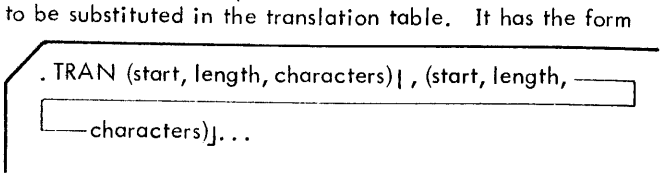
REM indicates that the device associated with the intermediate DCB upon which the final output file will be written is to be freed for use as the output file device. This allows a tape drive to be used for multiple purposes instead of tying it up until the end of the sort. REM applies only to a sequential sort (that is, sort with its intermediate files on magnetic tape).

DUMP informs Sort to dump its overlays in case of an abort.

TBUF,number specifies the number of buffers the Sort is to use for writing output string records from the tournament. The Sort normally uses two buffers when running in a stand-alone mode, linked mode, or co-resident mode. For larger sorts, two buffers are recommended. This option is valid when only using the Random technique. Valid values are 1 or 2. X'40' or X'F0' will default to 2; all other values are errors.

The PAGES, DCBS, REM, DUMP, and TBUF parameters may appear in any order on the LIMIT card.

TRAN This card specifies where certain characters are to be substituted in the translation table. It has the form



where

start is the beginning byte position in the translation table where characters are to be substituted. Start is expressed as up to a three-position numeric field.

length is the number of bytes, or characters, to be altered.

characters are the alternate characters to be substituted.

Each (start, length, characters) parameter must be contained on one card. That is, if a TRAN card is continued, the continuation card must start out with a period, then a space, and then a new (start, length, characters) parameter.

An example of the TRAN card is shown later in this chapter.

KEY TRANSLATION TABLE

If the collating sequence values of characters in one or more keys are to be translated (changed to other values), a key translation table with the following characteristics is constructed:

1. A 256-byte area is set up in high memory so as to contain the standard EBCDIC character set. The layout and function of this table is explained at the end of this section.

2. Key translation control records will be read from the control device until an end-of-data appears (this is applicable only to the program call mode of operation).

3. The "start" parameter in the TRAN card indicates the beginning location for character substitution.

4. The characters can consist of any of the valid characters in the Xerox EBCDIC collating sequence. These characters will replace characters in the translation table, beginning with the byte position indicated by the TRAN card "start" parameter.

5. Any characters in the standard table that are not replaced will be retained in the original EBCDIC collating sequence.

Table 1 is a table of the standard Xerox EBCDIC Collating Sequence. (Temporarily ignore the alternate characters above the characters "blank", A-Z, and 0-9; these pertain to an example that is explained below.) When this table is initially stored in memory, "Null" will occupy byte 1, "blank" will occupy byte 65, "A" will occupy byte 194, etc.

It is important to understand that the purpose of the table is not to provide Sort with a set of EBCDIC codes, but rather to indicate the relative position (i.e., rank) of each character in the sorting sequence. Initially, therefore, the letters A-Z (X'C1' - X'C9', X'D1' - X'D9', and X'E2' - X'E9') are lower in sequence value than the numerals 0-9 (X'F0' - X'F9').

The sequence value of any character ($character_x$) can be changed by replacing it with $character_y$, the character whose sequence value $character_x$ is to assume in the modified EBCDIC collating sequence. For example, to reorder the standard EBCDIC collating sequence so that the letters A-I would sort after the letters J-Z and the numbers 0-9, the user would enter new sequence (character) values into the positions originally occupied by these characters in the EBCDIC table. The new value that would be entered for each character in this example is shown above the character in Table 1. The table shows that card code 12-0 is to have the sequence value formerly associated with numeral 0, the letter A to have the sequence value formerly associated with numeral 1, the letter B the numeral 2, and so on.

The format for the TRAN card required to accomplish this transformation would appear as shown in Example 1.

The translation table is a byte table. It is entered by the Sort processor, using the EBCDIC code of a character as an index. In an unmodified table, the indexed byte contains its own index. For example, capital letter C, with an EBCDIC code of X'C3', is the $C3^{th}$ entry in the standard collating sequence; the $C3^{th}$ byte of the table contains X'C3'.

To modify the collating sequence, by using the TRAN option on the .KEYS card and using the .TRAN card, the user should insert a new index, or sequential collating position, into the translation table's byte for the affected character. For example, if the user wanted the letter C to collate after a blank (X'40'), but before any other of the special alphabetic or numeric characters (X'4A' through X'F9'), he could use the following .TRAN card:

```
.TRAN  (196, 1, 12)
                0
                9
                1
```

where

196 is the byte location (decimal) for the letter C (X'C3').

1 is the number of bytes to be altered.

12
0
9
1 (multipunch) is the card code for EBCDIC X'41'. This supplies a new collating sequence for the letter C, just following the blank (X'40').

Table 1. EBCDIC Collating Sequences

Byte	Most Significant Digit	Least Significant Digit																
		Hex. Value	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1		0	Null					HT			EOM							
17		1						NL										
33		2	ds	ss	fs	si												
49		3																
65		4	11-0 blank										¢	.	<	(	+	|
81		5	&										!	$	*	)	;	¬
97		6	-	/										,	%	–	>	?
113		7											:	#	@	'	=	"
129		8		a	b	c	d	e	f	g	h	i						
145		9		j	k	l	m	n	o	p	q	r						
161		A			s	t	u	v	w	x	y	z						
177		B																
193		C	(12-0)	J A	K B	L C	M D	N E	O F	P G	Q H	R I						
209		D	(11-0)	S J	T K	U L	V M	W N	X O	Y P	Z Q	0 R						
225		E	(0-8-2)	1	2 S	3 T	4 U	5 V	6 W	7 X	8 Y	9 Z						
241		F	12-0 0	A 1	B 2	C 3	D 4	E 5	F 6	G 7	H 8	I 9						CD

Example 1. TRAN Card

```
5  . (241, 10,  ABCDEFGHI)
4  . (209, 10,  STUVWXYZ0), (225, 10,  123456789), ;
3  . (123, 6, :#@'="), (193, 10,  JKLMNOPQR), ;
2  . (91, 6, !$*);¬), (97, 2, -/), (108, 5, , %->?), ;
1  . TRAN (65, 1,  ), (75, 6, ¢. <(+|), (81, 1, &), ;
```

The most common requirement for use of the TRAN option is to sort numbers before the alphabetic characters. If there is no requirement to consider the brackets, braces, or backslash characters (X'B1 through X'B5') as being among the key sort characters, then the .TRAN card could appear as shown in Example 2.

If the .TRAN card is to be prepared from a terminal, the multipunches in Example 2 cause problems. Also, the equal collating level of brackets, braces, and numbers may not be acceptable. If either situation exists, a lengthier method may be used as follows:

```
.TRAN   (194,9,KLMNOPQRS),(210,9,TUVWXYZ01),;
.       (227,8,23456789),(241,10,ABCDEFGHIJ)
```

The resulting collating sequence will be:

- Unchanged for EBCDIC codes X'00' through X'BF'.
- The numbers 0-9.
- Alphabetic characters capital A through Z.

OPERATING IN A PROGRAM CALL MODE

Any program may call out and execute Sort one or more times during a given job step. The calling program must perform the following tasks each time Sort is to be executed:

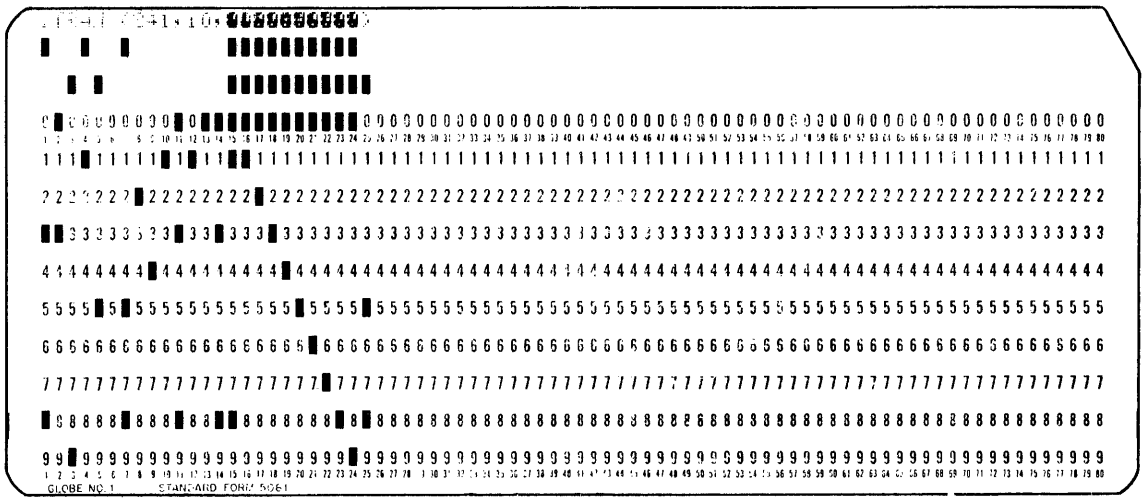
1. A page of common dynamic memory must be acquired by executing the procedure M:GCP 1.†
2. Sorting specifications must be set up in the first 240 bytes of the common memory. The format of the specifications can be determined by chaining together the three 80-byte job specification records shown later in this chapter in Figure 1.
3. The calling program may describe the characteristics of the input file by moving the input DCB into the common area immediately following the sorting specifications. If the input DCB is present in the common page, the output DCB must also be given.

   Note: The user's DCB can be modified at run time by an ASSIGN control card.
4. The calling program may describe the characteristics of the output file by moving the output DCB into the common memory area immediately following the input DCB.

   Note: The user's DCB also can be modified at run time by an ASSIGN control card.

†See the BPM/BP, RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.

Example 2. TRAN Card used to Sort Numbers before Alphabetic Characters



†Note that these represent numeric multipunch codes for X'B0' through X'B9'.

5. If the input file has a user label, the calling program must open the file, read the label, and verify it before calling Sort if the user wishes his header labels to be verified[†]. If the user does not wish to verify the header labels, the sorting specification must indicate a user-labeled file. The Sort program will read and bypass all input header labels. Trailer labels are not processed and should not be indicated in the sorting specifications.

6. If the output file has a user label, generate the label in the common memory area immediately following the output DCB. The first byte of the label information must contain the number of label characters that follow. This byte is not written on the output file. A binary nonzero byte following the output DCB indicates an output file header. If this option is to be used, both the input and the output DCBs must be specified in the common memory area.

   Note: The Sort specification indicating a user-labeled file does not apply to the output.

7. If the output file does not have a user label, place a binary zero in the first byte of common memory immediately following the output DCB.

8. The input and output files must be in a closed condition so that they may be accessed by Sort.[†]

9. Load the following registers:

   Register 6 contains the starting word address of common memory.

   Register 7 contains the starting word address of the input DCB in common memory. If no DCBs are contained in common memory, then register 7 must contain zero.

   Note: Both DCBs must be specified if register 7 is not zero. In addition, if register 7 is set to zero, it is not possible to pass an output user label to the Sort.

10. Execute an M:LINK procedure to Sort. This procedure causes the called program to commence operation. The following parameters are specified:

    a. Load module name = SORT.

    b. Account number = :SYS.

    Note: The password is not used.

The above procedure results in a checkpoint being performed on the calling program and initiation of the Sort program. The sorting specifications in the first 240 bytes of the common memory will be moved into the area normally filled by the Sort specification records used in the batch sorting procedure. The input and output DCB images in the common memory will be analyzed for file characteristics. Modification of F:SORTIN and F:SORTOUT will occur as if ASSIGN control commands had been processed as defined in Chapter 2. At the completion of Sort, control will be returned to the calling module at the location immediately following the M:LINK[†] procedure.

At the time of reentry, the files specified in the input and output DCBs will be closed. The common dynamic memory (one page) will be released by SORT via M:FCP.[†] Table 2 indicates register status.

Table 2. Register Status

Register	Status	
1-5	Unchanged	
6	Completion Codes	
	Code	Explanation
	0	Sort successfully completed
	1	Sort error – in and out record count out of balance
	2	Sort aborted – input/output error
	3	Sort aborted – specification error
	4	Sort aborted – registers give reason
	5	Sort aborted – memory overflow
	6	Sort aborted – illegal own-code action request.
	7	Not used
	8	Sort aborted – illegal decimal key
	9	Sort error – sequence error in output file
7	Number of records in the sorted file (hexadecimal value)	
8-9	For monitor use	
10-15	Unchanged	

If the completion code indicates an unsuccessful Sort, the calling program can either abort or enter an error procedure associated with the Sort call. No reentry into Sort at the point of error detection is possible.

Figure 1 shows the structure of the first 240 bytes of the common page in memory. The following paragraphs describe the contents of the common memory control fields.

[†]See the BPM/BP, RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.

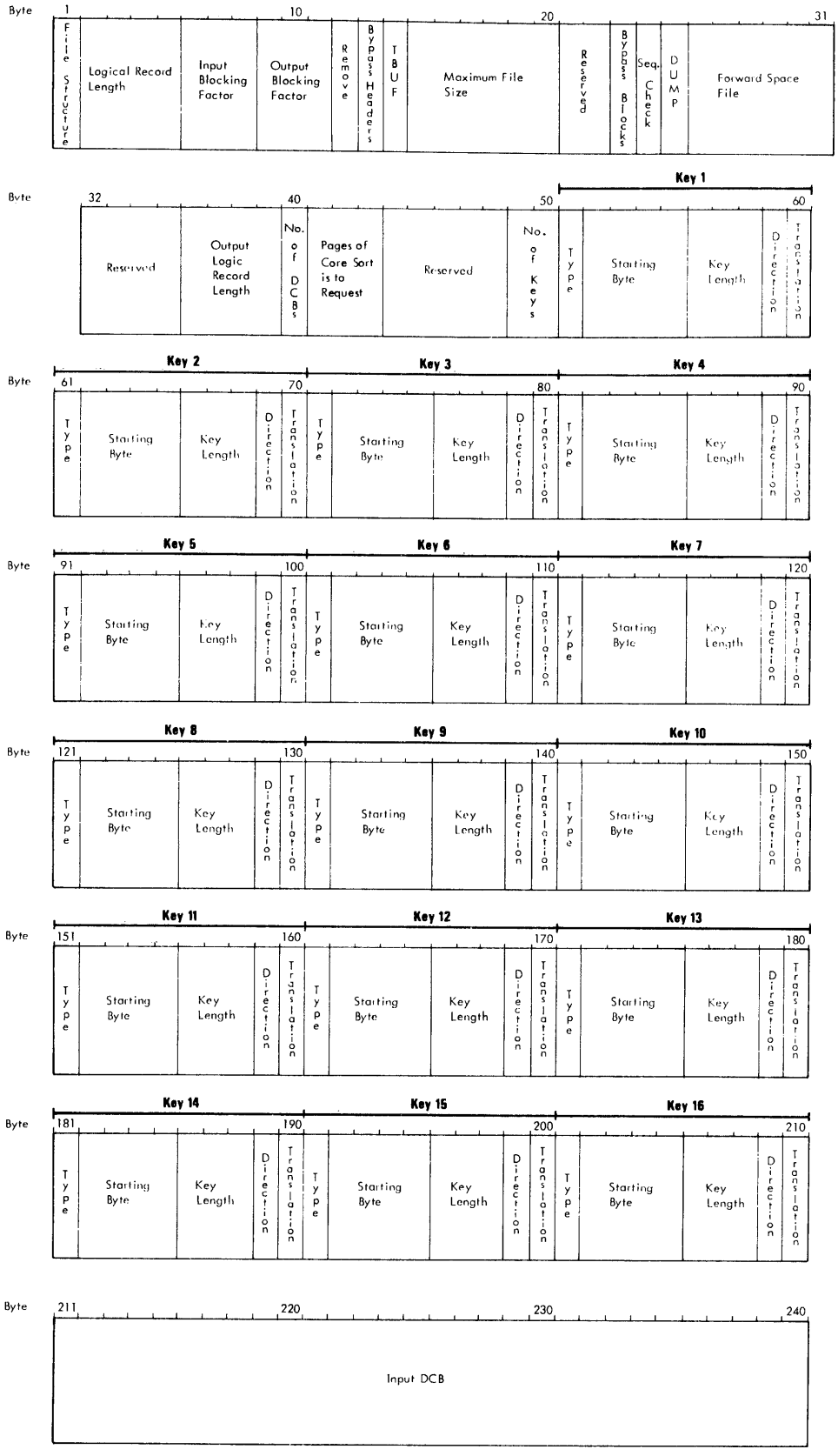


Figure 1. Common Memory Sort Specifications Area

Byte 1: Input file structure (required)

A means ANSI-formatted file.

M means monitor-formatted file.

U means user-formatted file with optional single-block header labels.

F means user-formatted file with multiple-block header labels.

Bytes 2-5: Logical record length (required)

xxxx specifies the four-digit logical record length in bytes.

For variable-length records, this is the maximum record length.

Bytes 6-8: Input blocking factor. Required for user-formatted files that are blocked, or for ANS blocked (F FORMAT) where the BLKL field is not provided on the !ASSIGN card for F:SORTIN.

xxx specifies the three-digit number of logical records per input block.

Note: Logical record length multiplied by the input blocking factor defines the size of the input buffer.

Bytes 9-11: Output blocking factor. Required for user-formatted files that are blocked, or for ANS blocked (F FORMAT) where the BLKL field is not provided on the !ASSIGN card for F:SORTOUT.

xxx specifies the three-digit number of logical records per output block.

Note: When the number of logical records in the last block is less than the blocking factor specified, a "short block" is written. No provision is made for padding a short final block.

Byte 12: Remove option

R means that the device associated with the intermediate DCB upon which the final output file will be written is to be freed for use as the output file device. This allows a tape drive to be used during a sort instead of tying it up until the end of the sort.

Anything else means that an entire tape drive will be reserved throughout a sort and will be used only as the output file device.

Note: This byte applies only when intermediate files are on sequential media (i.e., magnetic tape or nonrandom disk); it will be ignored when intermediate files are on random media (i.e., disk or RAD).

Byte 13: Bypass header labels

F means skip past the label file to the data file. Digits 1 through 9 mean skip over the specified number of label records. A blank means there are no header labels to be bypassed. Anything else causes an abort.

Byte 14: TBUF specifies the number of buffers the Sort is to use for writing output string records from the tournament. The Sort normally uses two buffers when running in a stand-alone mode, linked mode, or co-resident mode. For larger sorts, two buffers are recommended. This option is valid when only using the Random technique. Valid values are 1 or 2. X'40' or X'F0' will default to 2; all other values are errors.

Bytes 15-20: Maximum file size (optional)

xxxxxx specifies the six-digit maximum number of logical records to be accepted by Sort. If this parameter is zero or blank, the entire file (delimited by a file mark) will be input to Sort.

Note: Together with the forward space file parameter, this parameter allows the user to sort a file in successive slices. This procedure should be used to avoid intermediate file continuation reels or RAD/disk pack overflow when intermediate storage is assigned to RAD/disk packs. The sorted slices can then be merged into an output file.

Bytes 21-22: Reserved

Byte 23: Bypass unreadable blocks (optional)

Blank means abort.

Zero means bypass all unreadable record blocks.

Note: All unreadable blocks are shown on the user's listing log.

Byte 24: Sequence check output (optional)

S means sequence check on final output file required.

Anything else means suppress sequence checking.

Byte 25: DUMP (optional)

D means dump Sort overlays in case of an error abort.

Bytes 26-31: Forward space file (optional)

xxxxxx specifies the six-digit number of data records in the input file to be bypassed before sorting begins.

Note: The indicated number of logical records to be bypassed should exclude any user label records. This parameter permits the sorting of very large tape files in successive "slices", thus allowing the user to segment very large files and eliminate the need for intermediate tape continuation reels with the attendant waste of positioning time and the potential error involved in operator intervention (mounting and dismounting intermediate continuation reels). It will also prevent the occurrence of RAD/disk pack overflow.

Bytes 32-35: Reserved

Bytes 36-39: Output logical record length (optional)

xxxx specifies the four-digit logical record length in bytes that will be used during sorting and that will be written as output. This field is necessary if the output records are fixed length and the input records are variable length. If the output record length is greater than the input record length, unpredictable results can occur.

Note: If not present, input logical record length is used.

Byte 40: Number of intermediate DCBs assigned (optional)

x specifies the number of intermediate DCBs available to Sort; from 3 to 17 DCBs can be specified, expressed in hexadecimal form (that is, x can be a value from 3 to 9 or A to H). If byte 40 is left blank, six DCBs are assumed. When using the random sort technique, no fewer than six DCBs may be specified. See Chapter 4 for examples of its use.

Bytes 41-43: Number of 512-word pages of memory Sort is to request.

xxx specifies a three-digit number. If not present, all of memory will be requested.

Bytes 44-48: Reserved

Bytes 49-50: Number of Sort keys (required)

xx specifies the number of Sort key fields (16 maximum).

Byte 51: Key 1 data type (required)

A means alphanumeric (including unsigned zoned decimal).

B means binary (including normalized floating-point numbers).

P means packed decimal.

Z means zoned decimal. (If field is unsigned, specify A for a speed advantage.)

Bytes 52-55: Key 1 starting length (required)

xxxx specifies the four-digit starting byte location of this key, relative to the beginning of the logical record. The first (leftmost) byte in a record is byte 0001, the second is 0002, etc. All keys must start and end on byte boundaries.

Bytes 56-58: Key 1 byte length (required)

xxx specifies the three-digit length of the key field in bytes.

Byte 59: Direction of Sort on key 1 (assumed to be A unless otherwise specified)

D means descending sequence.

A (or anything else but D) means ascending sequence.

Byte 60: Key 1 translation (optional)

T means translate the input key field according to user-supplied character sequence values (read in from the control device) before sorting (applies only to alphanumeric and decimal keys).

A means translate to absolute value before sorting (applies only to binary keys. Signs are ignored. If a field has a high-order bit on, and no other bits are on, a fixed-point overflow will occur.

Byte 61: Key 2 data type (optional)

See key 1 definition, byte 51.
:
(repeat as required through key 16).

The Sort specification can terminate at byte 60; or if there are more keys, they can terminate at byte 210. Following the last key field, the user should specify input DCB, output DCB, user output header, label, and user output trailer label, if applicable, in that order. User header and trailer labels are always preceded by a byte that gives the length of the label; this byte is not written out.

SORT EXECUTION MESSAGES

Sort prints the following accounting messages on the LL device at the conclusion of each call. The LL device is normally defaulted to the line printer but can be reassigned to null or to some other device if it is undesirable to have information printed on the line printer during a job.

SORT VERSION x00: release date
RANDOM
SEQUENTIAL
RECORDS IN TOURNAMENT:
NUMBER OF MERGE BUFFERS:
INTERMEDIATE BUFFER SIZE:
RECORDS INPUT:
RECORDS OUTPUT:
RECORDS INSERTED IN:
RECORDS INSERTED OUT:
RECORDS DELETED IN:
RECORDS DELETED OUT:

If the user does not include any own-code modules, the last four accounting messages will not be printed. These messages are described in detail in Chapter 5.

ON-LINE SORT

The Sort processor may be called by on-line (terminal) users under the CP-V operating system. For a simple example of such usage, see Example A-7 in Appendix A. For more information on general on-line operations, refer to the CP-V Time-Sharing User's Guide, 90 16 92.

2. SORT FILE CHARACTERISTICS

The input and output files are controlled through the DCBs named F:SORTIN and F:SORTOUT. The files are predefined as:

1. Consecutive
2. Sequential
3. In (or out)
4. Save
5. Forty error retries

The following information is required by CP-V or BPM to access the files for a specific Sort, and must be supplied by the user in ASSIGN control commands:

1. Keyword DEVICE, FILE, or LABEL, plus NAME.
2. SN (if applicable to F:SORTIN). A maximum of 12 input reel/disk pack serial numbers may be specified.

The following keyword functions also may be included on the ASSIGNs that modify F:SORTIN and F:SORTOUT:

1. PASS
2. REL (to release the input file)
3. READ or WRITE
4. EXPIRE
5. SN (if applicable to F:SORTOUT). A maximum of 12 output reel/disk pack serial numbers may be specified.
6. TRIES
7. VOL
8. BIN, BCD, FBCD, PACK, or UNPACK (as required).

The BLOCK card input and output specification parameters (user format) determine whether the Sort will deblock records received from the monitor or block the record on the output side. If input blocking is necessary (ANSI fixed length and no BLKL field is given via the !ASSIGN card into the DCB for F:SORTIN or F:SORTOUT, or user fixed/variable length, unblocked), then the appropriate entry in the BLOCK card must be given. If output blocking is different from input blocking, then the appropriate entry in the block card must be made. In addition, the correct monitor options must be given on the ASSIGN cards for F:SORTIN and F:SORTOUT.

Although a monitor-formatted file may be designated as user-formatted, the reverse is not true. That is, in no case may a user-formatted file be designated as monitor-formatted.

3. SORT SYSTEM STRUCTURE

The Sort system will configure itself differently, depending on how it is called. The basic sequential and random Sort techniques consist of a root and four overlays.[†] The functions of this basic unit are described below.

1. SROOT – the root segment of the program.

2. SPRE – Reads control cards and branches to the sequential or random processor, depending on intermediate storage assignments.

3. SSP or SRP[††] – the root segment of the sequential or random technique. It remains resident throughout the Sort operation. The following are contained in this segment:

   a. Common Control Information – Data that is used by all phases of the program to control the sorting process.

   b. Linkage Routines – Program instructions to call in the successive modules of the Sort, and to exit at the end of processing.

   c. Service Routines – Program instructions to perform logical I/O, to move records, and to compare key fields.

4. SSP0 or SRP0[††] – overlay one of the program. This segment is called by a linkage routine to locate, check, and encode the sort specifications. It also allocates memory working areas and string blocking factors for use in subsequent phases.

5. SSP1 or SRP1[††] – overlay two of the program. This segment initially loads core with data in the sequential sort. For both random and sequential sorts, tournament tables are created along with the generation of the key comparison routines.

6. SSP2 or SRP2[††] – overlay three of the program. This segment performs an internal sort using the replacement-selection tournament technique and writes ordered strings of records onto intermediate files, distributing the strings as necessary for a polyphase merge. In the random sort a merge is performed after every n - 1 strings have been distributed, where n is the number of work files assigned.

7. SSP3 or SRP3[††] – overlay four of the program. In the sequential sort, this segment reads records from each file in the set of intermediate files and merges the ordered strings of records to produce a single ordered file after successive merge phases. In the random sort, only the final n-way merge is performed by this segment.

When Sort is to operate in a stand-alone environment or is linked to by another program (for example, a COBOL sort with USING/GIVING), a set of additional modules exists. These modules, P01, P11, P31, contain all of the external input/output functions for the sort. There is also a preprocessor that reads the sort parameter cards and determines which Sort processor to call. It determines this by examining the contents of the intermediate work file DCBs; that is, are they assigned to tape or disk? If Sort is being linked to, the preprocessor does not read control parameters but merely determines by DCB analysis which Sort processor should be called.

A diagram of the structure of the Sort system is shown in Figure 2.

As mentioned previously, the COBOL user has the option of using the Sort processor in a stand-alone environment or in a co-resident environment. In the stand-alone environment, the COBOL object program is swapped in and out of memory as necessary and Sort is swapped in and out as necessary. In the co-resident environment, Sort resides in memory along with COBOL, thus eliminating the need for Sort to be swapped in and out. The co-resident Sort receives input records from COBOL and passes sorted output records back to COBOL, and this can save the user up to 75 percent of the Sort input/output time. To use Sort in a co-resident environment, the user must do the following:

1. Include SRTx as one of the options in the COBOL processor control command, where x is S or R indicating the sequential or random Sort is to be used (see COBOL operations manual).

2. Specify the names of the Sort processor modules at load time (via the LOAD and TREE commands), so that the COBOL object code and the Sort processor will be loaded

---

[†] Remember that the sequential Sort processor is used when intermediate work files are exclusively on sequential storage media (i.e., magnetic tape or nonrandom disk), and the random Sort processor is used when the storage media is exclusively random (i.e., random disk and RAD).

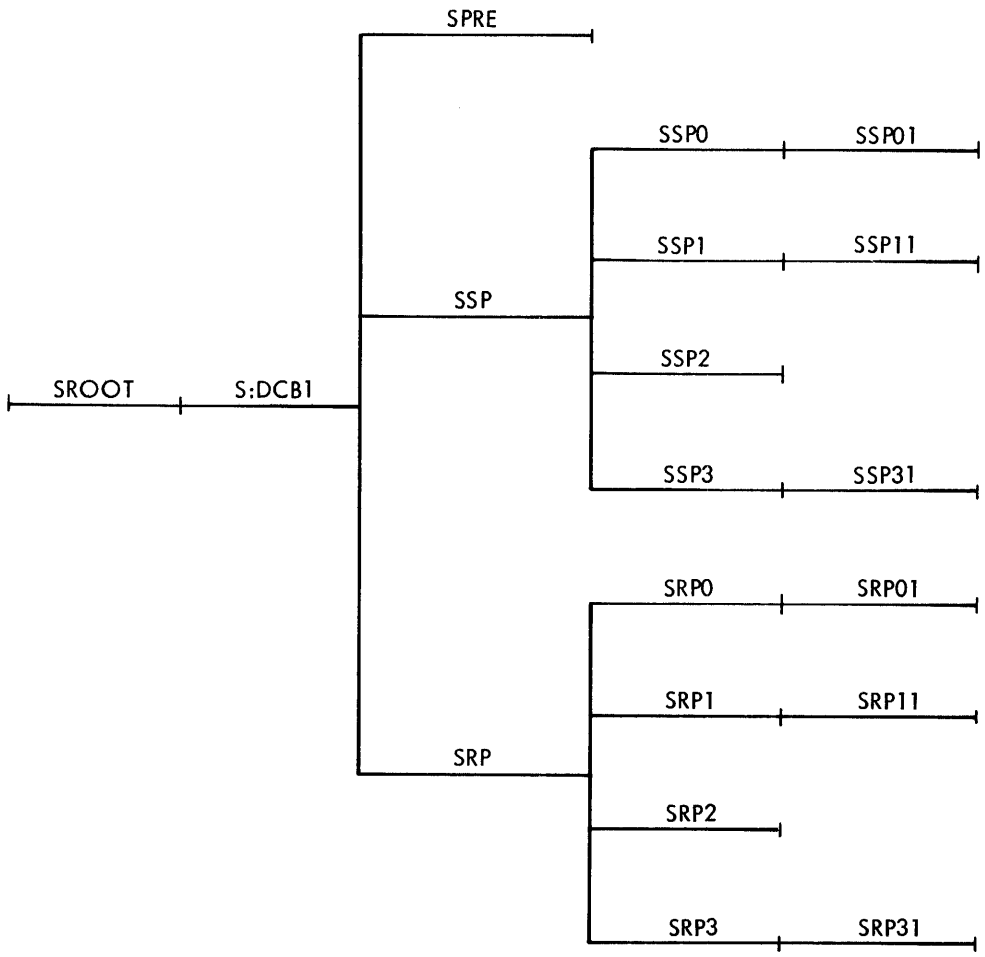
[††] The second letter of each name designates which processor is used – S refers to a sequential sort, and R refers to a random sort. Thus, for example, SSP references a sequential sort, and SRP references a random sort.

together to form one load module. It's up to the user to make sure that he gets the desired sort loaded properly; that is, he must specify whether the random sort or the sequential sort is to be loaded.

3. Assign intermediate files to random storage if the random sort has been called. The assignment technique is the same as for the stand-alone sort.

SAMPLE LOADS

The structure of the sort program must be transmitted to the monitor via the monitor control commands OVERLAY (or LOAD) and TREE. Examples of the OVERLAY and TREE commands for a stand-alone sort (with the structure of Figure 2) are shown in Example 3. More examples of these commands are shown in Appendix A. For examples of loading a co-resident sort, see the Xerox ANS COBOL (for BPM/CP-V)/OPS Reference Manual, 90 15 01.



Note: Co-Resident Sort: S:DCB1, SSP, SSP0, SSP1, SSP2, SSP3, SRP, SRP0, SRP1, SRP2, SRP3

Stand-Alone Sort called by ISORT or linked to by another program: All modules.

Figure 2. Sort System Structure

Example 3. OVERLAY and TREE Cards

```
1  !OVERLAY (EF,(SPRE),(SROOT),,;
2  !(SSP),(SSP0),(SSP01),(SSP1),,;
3  !(SSP11),(SSP2),(SSP3),(SSP31),(S:DCB1),,;
4  !(SRP),(SRP0),(SRP01),(SRP1),,;
5  !(SRP11),(SRP2),(SRP3),(SRP31)),,;
6  !(MAP),(PERM),(LMN,SORT),(BREF),(SL,F)
7  !TREE SROOT-S:DCB1-(SPRE,SSP-(SSP0-SSP01,,;
8  SSP1-SSP11,SSP2,SSP3-SSP31),SRP-(SRP0-SRP01,,;
9  SRP1-SRP11,SRP2,SRP3-SRP31))
```

Note: In this example it is assumed that all sort modules are in the same account under which the job is running. Hence, no account qualification is needed for each sort module. If the job were running under a user account and the sort modules were in another account (say, SYSGEN), then the sort modules in these two examples would have to be qualified. For example, (SSP) would become (SSP, :SYSGEN); (SSP0) would become (SSP0, :SYSGEN); and so on.

4. SORT USE OF INTERMEDIATE STORAGE

SEQUENTIAL SORTS

The sequential Sort processor is used when intermediate files are on magnetic tape, nonrandom disk, or both. This processor can use up to 17 DCBs to control its intermediate files. These DCBs default to public storage but they may be assigned to any appropriate input/output device via an ASSIGN control command.[†] The DCBs are designated F:SCRF1 through F:SCRF17.

The Sort attempts to write through F:SCRF2 first, then through F:SCRF3, and so on through F:SCRF17. F:SCRF1 is written after F:SCRF17. F:SCRF1 is the first output DCB used during the merge phase of the sort (i.e., Phase III). If the user wants to perform a sort, he should include ASSIGN control commands in his job, assigning the first n DCBs to tapes (where n can be an integer in the range from 3 to 17). In addition, he should supply the number, n, in the DCBS parameter of the Sort LIMIT control card (if he is using the job mode of operation) or in byte 40 of the common page (if he is using the program call mode of operation) – see Chapter 1. Sort will now use only the first n DCBs for its intermediate files. F:SCRF1 will be needed for the output file or F:SCRFn if a premature final pass is necessary (where n is either the DCBS value of the Sort LIMIT control card or the value in byte 40 of the common page). F:SCRFn will be used for the input file.

When the user assigns the intermediate files to magnetic tape, the file should always be assigned as a DEVICE, never as a LABEL.

By assigning all of the DCBs to specific devices and by not supplying the number-of-DCBs parameter in the Sort LIMIT control card or in byte 40 of the common page, the user will cause the Sort to ask the monitor for devices until the monitor replies that no more devices are available. When this reply is received, Sort adjusts its control tables to reflect the actual number of devices available. The user should note that when he is operating in this mode, Sort will reserve all available devices (up to 17) so that no device will be available to another job.

Note: If no DCBS parameter is given in the Sort LIMIT card or if column 40 of the common page is blank, Sort will default to six intermediate DCBs.

If the sequential Sort processor is used with disks sequentially, a significant amount of storage space must be provided.

The sequential Sort processor will operate using continuation intermediate reels. This sorting procedure is cumbersome because of the operator intervention required, and it should be avoided. File slicing, using the FILE card (SKIP and SORT parameters), is recommended for these cases.

[†] See the BPM/BP, RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.

An example is shown in Appendix A of the use of the sequential Sort processor, with three intermediate DCB files to magnetic tape.

RANDOM SORTS

The random Sort processor is used when intermediate files are on random disk, RAD, or both. This processor can also use up to 17 DCBs to control its intermediate files. In any event, a random sort must have at least six DCBs for its operation. This number provides nominal and acceptable sorting efficiency for the criss-cross (oscillating) sorting and merging technique.

Intermediate DCB assignments are made using the ASSIGN control command. The DCBs have the fixed identification F:SCRF1 through F:SCRF17. In addition, for a sort run the user should include either the DCBS parameter of the Sort LIMIT control card or byte 40 of the common page.

Note: If no DCBS parameter is given in the Sort LIMIT card or if column 40 of the common page is blank, Sort will default to six intermediate DCBs.

The criss-cross (oscillating) technique used by the random Sort processor does not allow any of the intermediate files to be freed for the final sort output as is possible with the sequential Sort processor.

Random sort is sensitive to improvements in execution configuration. Efficiency of the sort increases with the number of devices available. Separate channels for the disk packs are desirable. Assigning more than the basic six DCBs provides no real performance benefit unless the additional DCBs can be assigned to other devices.

The random Sort processor uses the random-access file capabilities provided by CP-V and BPM for accessing its intermediate work files. The total file space required by the user for intermediate sort work files must be 1.3 times the input file. The space for each intermediate work file is approximated by the following formula:

1. Convert file size to granules.
2. Multiply by 1.3.
3. Divide by the number of intermediate DCBs.
4. Use the resulting number as the RSTORE parameter in the ASSIGN command for each DCB.
5. The user may elect to divide the above value by 5 or 10, thus causing Sort to do intermediate file extensions and increasing Sort's chances of getting contiguous granule space.

This kind of storage is most conveniently provided with private disk packs specifically allocated for use as sort scratch packs.

An example of the use of the random Sort processor with six private scratch packs is shown in Appendix A.

5. SORT MESSAGES

The Sort messages printed in Table 3 are printed via the M:LL DCB.[†] Included in this table are error messages and general information messages (including the accounting messages printed at the end of each call to the Sort processor). Error messages usually cause a job abort, either immediately or after all remaining Sort parameters have been scanned for errors. The time of job abort is included in the description of each error message.

[†] See the BPM/BP, RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.

Table 3. Sort Error and Information Messages

Message	Comments
ANSI DECIMAL OR UNSPECIFIED FORMAT	An ANSI decimal format tape has been assigned to F:SORTIN or F:SORTOUT (Sort does not support this type of organization), or no format type has been given in an ANSI header or on an ASSIGN card. This error condition causes an immediate abort.
BLOCK DROPPED	An irrecoverable read error has occurred. The DROP parameter on the BLOCK card indicates that read errors are to be bypassed; or if using the common page method instead of Sort parameter control cards, byte 23 determines bypassing of read errors (see Chapter 1). The first 132 characters of the block will be listed unless the error occurs while reading an intermediate file, in which case the entire block will be printed. Sort will continue.
BLOCK LENGTH ABNORMAL	For blocked user-formatted files only, a physical block has been accessed that is shorter than the specified block length (logical record length times input blocking) and is not an integer multiple of the logical record length. This error condition causes an immediate abort.
DCBS PARAMETER NOT VALID	The specification for number of DCBs is invalid. This means that there is an error either in the DCBS parameter of the Sort LIMIT card or in byte 40 of the common page (see Chapter 1). An abort occurs after all remaining Sort parameters have been scanned for errors.
DISK SATURATED OR UNABLE TO SWITCH TO NEXT VOLUME	Sort was given an I/O error 42, 56, 01, or 57 by the monitor when trying to write an intermediate or output file. The probable cause is insufficient RAD storage for intermediate files. Storage for at least three times input file size is needed for intermediate files on the sequential sort and approximately 1.3 times the output file size for the random sort. This error condition causes an immediate abort.
FORWARD SP FILE	Invalid decimal digit specification has been found either in the SKIP parameter of the Sort FILE card or in bytes 26-31 of the common page. An abort occurs after all remaining sort parameters have been scanned for errors.
HDR FIELD NOT BLANK, F, OR 1-9	Either the HDR parameter of the Sort FILE control card or byte 13 of the common page is invalid (see Chapter 1). This error condition causes an immediate abort.
I/O ERROR CODE: xx	where xx is a code received from the monitor. Sort was given an I/O error while reading or writing. This error condition causes an immediate abort. If the code is 07 or 41, the user can elect to bypass these errors by specifying the DROP

Table 3. Sort Error and Information Messages (cont.)

Message	Comments
I/O ERROR CODE: xx (cont.)	parameter in the Sort BLOCK control card or by making the appropriate entry in byte 23 of the common page.
ILLEGAL DECIMAL KEY	An illegal decimal digit has been encountered in a zoned or packed decimal key. This error condition causes an immediate abort.
ILLEGAL OWN-CODE ACTION REQUEST	The user has returned from S:INUSO or S:OUSO via register 7, but byte 0 of register 6 does not contain a valid operation code (00, 01, or 02). This error condition causes an immediate abort.
IN - OUT	Gives, in hexadecimal, the input and output count of records at abort time.
IN-OUT COUNT (WITH DELETE) NOT EQUAL	Records have been "lost". The output file has been closed and saved, and may be used in a subsequent job. This error condition causes an immediate abort.
IN/OUT BLOCKING	An invalid decimal digit has been found in the input or output blocking factor specified in the BLOCK card or in bytes 6-8 or 9-11 of the common page (see Chapter 1). An abort occurs after all remaining Sort parameters have been scanned for errors.
INPUT OR OUTPUT FILE ALREADY OPEN	Can only occur if the user is performing his own I/O in own-code. This error condition causes an immediate abort.
INPUT RECORD LENGTH	An invalid digit has been found in the input record length specified in the Sort REC card or in bytes 2 through 5 of the common page (see Chapter 1). An abort occurs after all remaining Sort parameters have been scanned for errors.
INSUFFICIENT INFORMATION IN INPUT OR OUTPUT DCB	Sort was given an I/O error 01, 02, 03, 08, 09, 14, 30, 44, 46, or 47 when opening F:SORTIN or F:SORTOUT. Check ASSIGN cards or, if Sort is called as a subroutine, check the DCBs in the common page. This error condition causes an immediate abort.
INTERMEDIATE BUFFER SIZE: xxxxxx	This message indicates the intermediate buffer size, where xxxxxx is the size expressed in bytes. It is one of the accounting messages printed at the end of each call to the Sort processor.
INVALID CORE SIZE SPEC	An invalid digit has been found in the PAGES parameter of the Sort LIMIT card or in bytes 41-43 of the common page of a linked sort. An abort occurs after all remaining Sort parameters have been scanned.
INVALID TBUF VALUE	Random technique only. The TBUF value in position 14 of the common page, or on the .LIMIT card, has an incorrect digit. The acceptable values are 1 or zero (X'F0') or blank or 2. The last three characters default to 2.
KEY xx: ABSVAL	Translate to absolute value was specified for key xx, but this key is not binary. An abort occurs after all remaining Sort parameters have been scanned for errors.
KEY xx: DATA TYPE	Key xx data type is not specified as one of the valid types: AN, PD, ZD, BN, or BA (or Z, P, A, or B in the common page specifications). An abort occurs after all remaining Sort parameters have been scanned for errors.

Table 3. Sort Error and Information Messages (cont.)

Message	Comments
KEY xx: LENGTH	The length specification for key xx contains an invalid digit, or key xx length plus starting position is greater than the logical record length, or key xx length is greater than the maximum allowed for that key's data type. An abort occurs after all remaining Sort parameters have been scanned for errors.
KEY xx: START	Key xx start specification has an invalid digit, or key xx starts outside the record. An abort occurs after all remaining Sort parameters have been scanned for errors.
KEY xx: TRANSLATION ERROR	Translation was specified for key xx, which is binary. An abort occurs after all remaining Sort parameters have been scanned for errors.
MEMORY OVERFLOW	A common page is not available for storage of Sort specifications, or insufficient core storage is available for the INPUT/OUTPUT buffers and four records. This error condition causes an immediate abort.
NOT 1--16 KEYS	The number of key fields exceeds 16. An abort occurs after all remaining Sort parameters have been scanned for errors.
NOT 3--17 DEVICES/FILES AVAILABLE	The minimum number of devices or public storage files (dependent on user assignments) is not available. The minimum number is three for a sequential sort, or six for a random sort. This error condition causes an immediate abort.
NUMBER OF FILE EXTENTS FOR DCB ßßßßßßß:ß0FßßßßßßßßGRANULES EACH	This is an accounting message that tells the random Sort user that Sort has opened up to 20 temporary scratch files for each specified DCB. This is because the RSTORE value as originally requested was too low to complete the job.
NUMBER OF KEYS INVALID	An invalid digit has been found in bytes 49-50 of the common page in a linked sort. This error condition causes an immediate abort.
NUMBER OF MERGE BUFFERS: xxxx	This message indicates the number of buffers involved in a merge. It is one of the accounting messages printed at the end of each call to the Sort processor.
OPERATOR ERRORED OR ABORTED JOB	The Sort has been intentionally aborted via the control device.
OUTPUT RECORD LENGTH	An invalid digit has been specified as the output record length in the Sort REC card or in bytes 36-39 of the common page (see Chapter 1). An abort occurs after all remaining Sort parameters have been scanned for errors.
PARAMETER VALUE LESS THAN THE LEGAL MINIMUM VALUE	One of the following parameters is less than 1 (probably blank) when a value is required. Logical record length (specified in the Sort REC card or in bytes 2-5 of the common page – see Chapter 1). Any key starting position or key length (specified in the Sort KEYS card or in bytes 52-58, 62-68, etc., of the common page – see Chapter 1). An abort occurs after all remaining Sort parameters have been scanned for errors.

Table 3. Sort Error and Information Messages (cont.)

Message	Comments
RANDOM	This message indicates that the random Sort processor has been called. (In other words, intermediate files are stored on disk or RAD.) It is one of the accounting messages at the end of each call to the random Sort processor.
RECORD LENGTH ABNORMAL	A record has been read that is shorter than the highest key position. Register 8 has length of record read. This error condition causes an immediate abort.
RECORDS DELETED IN: xxxxxx	This is an accounting message that indicates the number of input records deleted via user own-code. For example, suppose there is a total of 10,000 records in an input file. If the user deletes 1000 of these via user own-code, the following two messages will appear in the accounting messages printed at the end of each call to the Sort processor: RECORDS INPUT: 9000 RECORDS DELETED IN: 1000
RECORDS DELETED OUT: xxxxxx	This is an accounting message that indicates the number of output records deleted via user own-code (that is, the number of records that made it through the intermediate sort but were deleted before being placed in the output file). For example, suppose 300 records made it through the intermediate sort. If the user deletes 20 of these via user own-code, the following messages will appear in the accounting messages printed at the end of each call to the Sort processor: RECORDS OUTPUT: 280 RECORDS DELETED OUT: 20
RECORDS IN TOURNAMENT: xxxxxx	This message indicates the number of logical records involved in the internal sort. It is one of the accounting messages printed at the end of each call to the Sort processor.
RECORDS INPUT: xxxxxx	This message indicates the number of logical records input to the sort. (The number does not include records in the input file that may have been deleted via user own-code – see RECORDS DELETED IN:, above.) This is one of the accounting messages printed at the end of each call to the Sort processor.
RECORDS INSERTED IN:	This is an accounting message that indicates the number of input records inserted via user own-code. For example, suppose there is a total of 10,000 records in an input file. If the user inserts via own-code 934 records, the following two messages will appear in the accounting messages printed at the end of each call to the Sort processor: RECORDS INPUT: 10934 RECORDS INSERTED IN: 934
RECORDS INSERTED OUT:	This is an accounting message that indicates the number of output records inserted via user own-code. That is, the number of records that made it through the intermediate Sort phase and were placed in the output file along with inserts. For example, suppose 300 records made it through the intermediate Sort. If the user inserts 20 more records via S:OUSO own-code, the

Table 3. Sort Error and Information Messages (cont.)

Message	Comments
RECORDS INSERTED OUT: (cont.)	following messages will appear in the accounting messages printed at the end of each call to the Sort processor: RECORDS OUTPUT: 320 RECORDS INSERTED OUT: 20
RECORDS OUTPUT: xxxxxx	This message indicates the number of logical records output, and is one of the accounting messages printed at the end of each call to the Sort processor. (The count in this message does not necessarily reflect the number of logical records that went through the intermediate sort. To find this number, add the values in the two accounting messages RECORDS OUTPUT and RECORDS DELETED OUT.)
REG 9 DEFINES LIMIT	Some user limit was exceeded while running the Sort job. In Register 9 will be found a bit configuration that defines the type of limit exceeded. The bit configuration is described in the CP-V Batch Processing Reference Manual, Publication 90 17 64, under the M:XCON description.
RSTORE VALUE TOO SMALL FOR I/O	A user-supplied RSTORE value for an intermediate DCB used by the RANDOM processor was smaller than that required for a write operation. This causes an immediate abort. Either increase RSTORE or reduce the number of intermediate DCBs.
SEQUENCE ERROR IN OUTPUT FILE	The user has caused a sequence error in his output own-code. This error condition causes an immediate abort.
SEQUENTIAL	This message indicates that the sequential Sort processor has been called. (In other words, intermediate files are stored on magnetic tape or sequentially on disk.) It is one of the accounting messages printed at the end of each call to the sequential Sort processor.
SLICE SIZE	An invalid digit has been encountered in the Sort parameter of the Sort FILE card or in bytes 15-20 of the common page. An abort occurs after all remaining Sort parameters have been scanned for errors.
SORT VERSION xxx: mm-dd-yy	This message indicates the current version of Sort and its release date, where xxx represents the version and mm-dd-yy represents the release date. For example, SORT VERSION F00: 04-15-75 means that the current version of Sort is F00 and that the release date is April 15, 1975. This is one of the accounting messages printed at the end of each call to the Sort processor.
SPECIFICATION ERROR	This error message is always printed in combination with one or more other error messages. It indicates that one or more specification errors have been detected, as indicated by the messages that precede it.
STRING LABEL ERROR	This message is caused by hardware or tape drive problems. It causes an immediate abort.

Table 3. Sort Error and Information Messages (cont.)

Message	Comments
TRANSLATION TABLE LOCATION ERROR	The start parameter on the Sort TRAN card contains a value greater than 256, or the number of characters in the TRAN card will extend past position 256 in the translation table. Either of these error conditions causes an immediate abort.

6. MERGE

INTRODUCTION

Besides sorting, another basic process in data manipulation is the merging of several files into one file. The Merge program provides this capability for a Xerox 560 or Sigma 5-9 computer with sufficient memory capacity to provide adequate program and working storage (see "Equipment Configuration" below). Merge operates under the Control Program-Five or Batch Processing Monitor systems. Since Merge is device-independent, a mixture of devices and files may be used. Parameters supplied at run time define the characteristics of each job.

FEATURES

Major features of the Merge program are summarized as follows:

1. A maximum of eight input files of the same format may be merged into one output file of another format.

2. Files can consist of either ANSI-, monitor-, or user-formatted records, but all input files must be of the same format.

3. Fixed-length records in blocked or unblocked format can be handled. (In monitor format they are unblocked to merge.) Variable-length records may be processed in blocked or unblocked ANSI format, and in unblocked monitor and user formats.

4. Merging can be accomplished on multiple key fields, in either an ascending or descending sequence.

5. Records containing from 1 to 16 key fields can be merged.

6. Merging can be performed on the following types of key field data:

   a. Alphanumeric

   b. Binary (including normalized floating-point numbers)

   c. Zoned decimal

   d. Packed decimal

7. User own-code can be inserted at specified points, permitting record modification, record deletion, and access to user header and trailer records.

8. User-specified character-collating sequence can be utilized.

9. Buffered input/output is used (core permitting).

EQUIPMENT CONFIGURATION

Merge is designed to operate efficiently in a minimum hardware environment. The minimum configuration consists of a system that uses Control Program-Five or the Batch Processing Monitor, and additionally provides sufficient core memory to provide approximately 5,000 words of storage for the program (exclusive of the resident monitor), peripheral storage to handle all files, and additional working storage for at least one block from each file.

These minimum requirements presume that the entire system's resources are dedicated to a single job. Additional components may be required to carry out other tasks simultaneously.

PROGRAM ORGANIZATION

The Merge program is carried in the system library as a single, unsegmented program. The program is subdivided into three routines as described below.

PH:RES

PH:RES preserves the registers and branches either to other routines or back to the monitor. It also outputs messages to the user as required.

PHASE I

Phase I acquires various user-supplied parameters (e.g., file characteristics, record characteristics, default options, etc.), checks them for consistency, and builds control tables for use by the following phases. This routine also determines the memory requirements for merging, and sets intermediate storage buffering factors and working storage values for the merge routine (Phase II).

PHASE II

If no errors were found in the previous routine, Phase II performs the merge. One record from each file is read in, sequenced, and output, thereby keeping memory requirements to a minimum and eliminating the need for intermediate storage.

OPERATING MODES

The user can call the Merge processor out of the system library as an independent processor in the batch job mode; he does this via the monitor MERGE control command. The user's specifications are then read from the control input device, and processing is initiated as explained below. When the Merge contains user own-code subroutines, it exists as a customized processor under a user account number.

FILE ORGANIZATION

Files that are to be merged consist of fixed- or variable-length logical records, and fall into three major categories: ANSI-, monitor-, and user-formatted. ANSI-formatted files may be fixed or variable length, blocked or unblocked. Monitor-formatted file records may be of fixed or variable length and are unblocked to merge. User-formatted files may be of fixed length and blocked, or of variable length and unblocked. The last physical record in the file may be shorter than a full block if the proper multiple of logical records is not present. Padding characters added by the user to fill out the last block will be treated as data and merged accordingly. Input and output files may be of a different format. Maximum record length must be the same for all files.

If file header labels are used, the user may provide program modules to process input and output headers; otherwise the labels will be skipped. If own-code modules are available, Merge will read the first block or logical record or will write the last block or logical record with a file mark if necessary. Any other header label input/output including positioning is the responsibility of the user.

The maximum number of records is limited only by the output storage facility. Releasing a file on an ASSIGN card does not free storage for use by Merge. General characteristics of monitor, ANSI, and user-formatted files are outlined below.

ANSI-FORMATTED FILES

The format and structure of ANSI-formatted files conform to the American National Standard for Magnetic Tape Information Interchange, except that the decimal tape format is not processed. The Block Header Field is restricted to 0-byte length for fixed-length format and to a length of four bytes for variable-length formats. If a Header 2 label (HDR2) is not present, appropriate file description information must be supplied via monitor ASSIGN cards (see the CP-V/BP Reference Manual, 90 17 64). Only one 80-byte user header label (UHL1) or trailer label (UTL1) is permitted per file volume. If user labels are present, the user must provide program instructions for checking input labels and creating output labels.

MONITOR-FORMATTED FILES

Monitor-formatted files have their packing structure determined by the monitor at the time of generation. The file structure cannot be changed by user specifications. The first record in the file contains the monitor label, which in turn, contains information such as the file name, user account number, and logical record blocking factor. If the file is on tape, an optional user label (255-byte maximum) is also present. The user label will be ignored unless the user explicitly indicates that "own-code" has been provided to process the label and to generate a new label for the merged file.

USER-FORMATTED FILES

User-formatted files have their packing structure explicitly controlled by the user. The Merge requires blocking factors from the user in order to provide automatic blocking and unblocking of logical records. The first physical block in the file may or may not be a user label. A single-block user header label may not be larger than 255 bytes. Multiple-block user header labels are one or more blocks that are not larger than 255 bytes, ending with a file mark. If a label is present, the user must provide program instructions for checking the input label and for generating the output label. The last data block may be a short block without padding characters. The end-of-file mark may be followed by a user trailer label.

If user trailer labels are specified, the user must make program modules available in the user library to process input and output trailers. Trailers appear at the end of each physical volume.

MERGE LOAD STRUCTURE

Included in the Merge program are various secondary references that must be linked to user own-code when headers, trailers, or access to data records are specified. The user assembles his program as independent relocatable object modules and then builds a new module (with his account and new load module name) which appends his program to Merge. This results in a unique program for that particular merging task.

The user's program must contain external definitions of entry points to be used as follows:

When Merge Specification Is	Merge Branches to User's Definition of	When
Header Labels	MINHED	After reading an input user header
	MOUHED	Before writing an output user header
Trailer Labels	MINTRL	After reading an input user trailer
	MOUTRL	Before writing an output user trailer
Input and/or Output Own-Code	MINUSO	After accessing each input data record
	MOUSO	Before outputting each merged data record

User own-code modules may include their own DCBs and input/output functions. Own-code modules with their DCBs remain resident during the entire job, with a resulting decrease in working storage available to Merge. A limitation exists in the number of open DCBs, depending on the user's SYSGEN.[t] User definitions are explained below.

Merge cannot protect the integrity of its input buffer or record areas during the execution of user own-code instructions. The user must take all precautions necessary to prevent his program from modifying areas outside his jurisdiction.

INPUT PHASE OWN-CODE

MINHED This external definition is assumed to exist in a user module appended to Merge whenever a user input header label is present. The user's program will be entered after each input file header is read by Merge. The starting location of an input buffer will be passed on and the own-code will be responsible for verifying its contents. If needed, the user may perform his own read-in of parameters (such as a data card) against which the input header can be compared. After verification, the user will return to Merge at a specified linkage point. If the file is ANSI- or monitor-formatted and MINUSO is specified, a logical record will be acquired and accessed before providing linkage to the next header. If user-formatted with MINUSO, the next physical block will be read and access will be given to the first record on that block before providing linkage to the next header. The user's header is truncated if its length exceeds 80 bytes (ANSI format), 255 bytes (monitor format), or that of the data block (user format).

MINTRL This external definition is assumed to exist in a user module appended to Merge when trailer records following each reel of the input file are present. The user's program will be entered after each input trailer is read by Merge.

At the time of entry to MINHED and MINTRL, the following conditions exist:

Register 4 contains the file number in binary (1-8).

Register 5 contains the return address (B *R5).[tt]

Register 6 contains the byte address of the Merge input buffer.

Input buffer byte 1 contains the length of the input header/trailer.

Input buffer byte 2 contains the first byte of the header/trailer record.

MINUSO This external definition is assumed to exist in a user-module appended to Merge when access to input data records is desired. The user's program will be entered after each input logical record is acquired. The starting location of the logical record will be passed on, and the user may perform any of the following operations on the data:

1. Summing, rearranging, coding and decoding, etc.
2. Record length expansion or reduction. The input logical record length parameter should be set at the expanded size. The record length should not be reduced to less than the key length.

After completion of the user's processing of each logical record, the own-code may return to Merge at one of two specified linkage points that either enter the record into the merging process or return to the input record read process, thus deleting the current logical record.

At the time of entry to MINUSO, the following conditions exist:

Register 4 contains the file number in binary (1-8).

Register 5 contains the return address to enter the last record into the merging process (B *R5).[tt]

Register 6 contains the byte address of the last record read.

Register 7 contains the return address to delete the last record (B *R7).[tt]

Register 8 contains the byte length of the record passed to the user and must contain the record length on return to Merge. This allows the user to change the record length within the maximum size specified.

OUTPUT PHASE OWN-CODE

MOUHED This external definition is assumed to exist in a user module appended to Merge when a user wishes to prepare an output file header label. The user's program will be entered before any data records are written on each physical volume of the output file. The user may construct his header in a work area contained within MOUHED and return to Merge with the byte address of that area. The first byte of the area contains the number of bytes in the header, followed by the header information itself. This first byte is not written to the output file.

MOUTRL This external definition is assumed to exist in a user module appended to Merge when a user wishes to prepare output trailer records following each volume of an output file. The user's program will be entered at the end of each volume. User processing follows the same rules as for header labels.

---

[t] See the BPM/BTM/SM Reference Manual, 90 17 41, or the CP-V/SM Reference Manual, 90 16 74, System Generation chapter, as appropriate.

[tt] The return is accomplished by executing a branch to the address in the appropriate register.

The following conditions are assumed to exist in connection with MOUHED and MOUTRL:

At entry, register 5 contains the return address (B *R5).[†]

At exit, register 6 contains the byte address of a user's output buffer (set by user).

Output buffer byte 1 contains the length of header or trailer record.

Output buffer byte 2 contains the first byte of header or trailer record.

MOUSO This external definition is assumed to exist in a user module appended to Merge when access to output data records is desired. The user's program will be entered before each logical record is to be disposed of. The starting location of the logical record will be passed on, and the user may perform summing, rearranging, coding, and decoding of the data. Record length modification is permitted. After completion of the user's processing of each logical record, the own-code may return to Merge at one of two specified linkage points that either sends the record to the merged file output process, or returns to the merging process, thus deleting the current logical record.

At the time of entry to MOUSO, the following conditions will exist:

Register 5 contains the return address to write the current record in the merged output file (B *R5).[†]

Register 6 contains the byte address of the current output record.

Register 7 contains the return address to delete the current record (B *R7).[†]

Register 8 contains the byte length of the record passed to the user and must contain the record length on return to Merge. This allows the user to change the record length within the maximum size specified as the output logical length.

KEY FIELDS

Records to be merged may contain from 1 to 16 key fields. Key fields must be contained within the fixed portion of a record. If keys extend outside the boundary of a variable record, the comparison will be unpredictable; an error may or may not be detected. A key field must be aligned on byte boundaries and may contain the following types of data:

Data Type	Length of Key Field
Alphanumeric	A maximum of 255 8-bit EBCDIC characters. Unsigned, zoned decimal fields should be placed in this category for increased efficiency in the comparison process.
Binary	A maximum of 8 bytes.
Packed decimal[††]	A maximum of 31 decimal digits, plus sign, packed into 16 bytes.
Zoned decimal[††]	A maximum of 31 decimal digits, plus sign, in EBCDIC form contained in 31 bytes.

Individual key fields may be merged in ascending or descending sequence. Comparison results are algebraic for decimal and binary keys, and absolute EBCDIC for alphanumeric keys. The collating sequence value of the characters used in alphanumeric, packed decimal, or zoned decimal key fields may be transformed according to a user-supplied table of sequence values before the key fields are merged. If character sequence translation is specified as a Merge parameter, the user must supply one or more key translation control cards showing the desired sequence changes to the standard EBCDIC character set.

MERGE PARAMETERS

Use of the Merge program requires submission of parameter control cards. The cards contain data necessary to set up the Merge process, such as key field definitions, file and record structure, and error option selections. Merging parameters are obtained from the last assigned control entry device.

The specific parameter control cards and their formats are defined below. These parameter control cards may appear in any sequence in the control deck following the MERGE command. All but the NOTE parameter control card may be used only once in the control deck; the NOTE card may be used as many times as required.

Each parameter control card must start with the following:

1. A period in column 1.
2. The name of the parameter control card (e.g., NOTE) beginning in column 2.
3. At least one blank following the name of the card.

[†]The return is accomplished by executing a branch to the address in the appropriate register.

[††]The user's system must have decimal arithmetic capability to handle this type of key field.

If a parameter control card requires more space than is available on a card, it can be continued on one or more following cards (permitted only with the KEYS and TRAN control parameters). When a parameter control card is to be continued on another card, the following rules apply:

1. Each card that is to be continued on another card must be terminated with a semicolon.

2. Each continuation card must have a period in column 1 and a blank in column 2.

REC This card specifies record length and whether or not record sequence errors in the merged output file are to be ignored. It has the form

```
.REC (input,output,NSEQ)
```

where

input specifies the input file record length, expressed in number of bytes.

output specifies the output file record length, expressed in number of bytes.

NSEQ indicates that sequence errors in the merged output file are to be ignored.

The user must always provide the input record length. The output record length is optional; if it is not specified, it is assumed to be the same as the input record length. In a variable-length file, record length is the maximum physical record length.

BLOCK This card specifies blocking factors for blocked files on tapes other than ANS labeled tapes; it is not needed for unblocked files. It has the form

```
.BLOCK (input,output,DROP)
```

where

input specifies the input blocking factor; that is, the number of records per block. The number can consist of up to three digits (999).

output specifies the output blocking factor; that is, the number of records per block. The number can consist of up to three digits (999).

DROP indicates that bad blocks are to be dropped. (A bad block is a block in which an I/O error is encountered during a read operation.) This option may appear anywhere within the parentheses.

The user must always specify the input blocking factor on blocked files. The output blocking factor is optional; if it is not specified, it is assumed to be the same as the input blocking factor.

FILE This card specifies the number of files to be merged and the header labels to be skipped; it is required for a merge operation if an end-of-file is to be written after the output user header. The form is

```
.FILE (FILES,number),(IHDR,x),(OHDR,x)
```

where

FILES,number specifies the number of files to be merged. The number can be an integer from 1 to 8 inclusive.

IHDR,x specifies the input header labels to be skipped. The x can consist of an F or any digit from 1 to 9. An F indicates to skip past the file of header labels to the data file. A digit indicates to skip past the indicated number (1-9) of header labels. If there are no input headers to be skipped, omit this parameter from the .FILE card.

OHDR,x specifies if merge is to write an end-of-file after the user's output header. If this is desired, x must be an F; anything else will be ignored. If there are no output headers to be skipped, omit this parameter from the .FILE card.

KEYS This card specifies the merge key characteristics; it is not required when only one input file is present. The form is

```
.KEYS (origin,length,type,direction,TRAN)
      [,(origin,length,type,direction,TRAN)]...
```

where

origin specifies the starting byte position in the record. The first byte is 1, the second byte is 2, etc.

length specifies the key length. Key length is expressed in number of bytes and can consist of up to three digits.

type indicates the merge key type and can be any of the following:

AN indicates alphanumeric.

PD indicates packed decimal.

ZD indicates zoned decimal.

BN indicates binary.

If type is omitted, alphanumeric type is assumed.

direction indicates the direction of the sort (A = ascending merge, D = descending merge). If direction is omitted, an ascending merge will be done.

TRAN indicates that translation table should be used with the key field.

NOTE This card allows the user to insert comments into the Merge parameter control deck. A NOTE card may not be continued onto other cards, but any number of NOTE cards may be used. This card has the form

```
.NOTE [any text]
```

TRAN This card specifies where certain characters are to be substituted in the translation table (see the section titled "Key Translation Table" in Chapter 1). It has the form

```
.TRAN (start,length,characters)[,(start,length,
      characters)]...
```

where

start is the beginning byte position in the translation table where characters are to start being substituted. Start is expressed as a three-position numeric field.

length is the number of bytes, or characters, to be altered.

characters are the alternate characters to be substituted.

Each (start, length, characters) specification must be contained on one card. That is, a specification cannot be broken up and continued to another card.

ON-LINE MERGE

The Merge processor may be called by on-line (terminal) users under the CP-V operating system. For a simple example of such usage, see Example B-4 in Appendix B. For more information on general on-line operations, refer to the CP-V Time-Sharing User's Guide, 90 16 92.

7. MERGE FILE CHARACTERISTICS

The input and output files are controlled through the DCBs named F:MRGOUT and F:MRGINn, where n is 1-8. The files are predefined as

1. Consecutive.
2. Sequential.
3. Save.
4. Forty error retries.

The following information is required by CP-V or BPM to access the files for a specific merge, and must be supplied by the user in ASSIGN control commands.

1. Keyword DEVICE, FILE, or LABEL, plus NAME.
2. SN, if applicable to F:MRGINn. A maximum of 12 input reel/disk pack serial numbers may be specified for each input file.
3. IN or OUT, when appropriate.

The following keyword functions also may be included on the ASSIGNs that modify F:MRGOUT and F:MRGINn:

1. PASS.
2. REL (to release the input file).
3. READ or WRITE.
4. SN, if applicable to F:MRGOUT. A maximum of 96 output reel/disk pack serial numbers may be specified.
5. TRIES.
6. VOL.
7. BIN, BCD, FBCD, PACK, or UNPACK (as required).
8. EXPIRE.

Values in the blocking factor specification parameters determine whether the merge will deblock records received from the monitor, and pack the records on the output side. If a blank blocking factor field is present, the merge will proceed with logic compatible with monitor-formatted files. Since determination of input and output is made separately, complete conversion freedom is allowed. However, all input files must be of the same format, and the record length must be the same for all.

Note: In no case may a user-formatted file be designated as monitor-formatted.

8. MERGE MESSAGES

The Merge messages printed in Table 4 are printed via the M:LL DCB.[†] Both error messages and general information messages are included in this table. An error message is usually preceded by the message MERGE ABORT or the message MERGE ERROR; these two messages indicate what happens after an error has been encountered, and are described along with the other messages in Table 4.

[†] See the BPM/BP, RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.

Table 4. Merge Error and Information Messages

Message	Comments
ANSI DECIMAL OR UNSPECIFIED FORMAT	An ANSI decimal format tape has been assigned to F:MRGINn or F:MRGOUT (Merge does not support this type of organization), or no format type has been given in an ANSI header or on an appropriate ASSIGN card. This error condition causes an immediate abort.
IHDR FIELD NOT BLANK, F, OR 1-9	See MERGE ERROR SPECIFICATION CARD FIELD, below.
INPUT BLOCKING	See MERGE ERROR SPECIFICATION CARD FIELD, below.
INPUT FILES NOT COMPATIBLE	The input files are mixed ANSI and user or monitor formats or there are mixed types (undefined, fixed or variable blocked) within the ANSI format.
KEY xx BOUNDARY	Key xx ends outside the boundaries of the record. Check the key start position and key length against total record length. This error will cause a job abort after all remaining Merge parameter control cards have been scanned for errors.
KEY xx DATA TYPE	The data type of key xx is not specified as one of the valid data types. Valid data types are AN, PD, ZD, BN, or BA. This error will cause a job abort after all remaining Merge parameter control cards have been scanned for errors.
KEY xx LENGTH	The specified length for key xx contains an invalid digit, or exceeds the maximum length permitted for its data type. This error will cause a job abort after all remaining Merge parameter control cards have been scanned for errors.
KEY xx START	The starting byte specification for key xx contains an invalid digit, or key xx starts outside the record. This error will cause a job abort after all remaining Merge parameter control cards have been scanned for errors.
LABEL I/O	An I/O error has been encountered during the read of a header or trailer label. This error condition causes an abort, but only after all the input files have been opened.
LOGICAL RECORD LENGTH	See MERGE ERROR SPECIFICATION CARD FIELD, below.
MERGE ABORT	This error message may be printed alone or in combination with other messages to indicate that an immediate abort has taken place. Other tasks within the job will be skipped.

Table 4. Merge Error and Information Messages (cont.)

Message	Comments
MERGE ABORT: PAGES REQ. : p AVAIL. : p	Not enough working storage is available to hold at least one block of data from each file. p specifies the number of 512-word pages of memory, designating both the required minimum and the actual available number. The user must reorganize the job into a number of tasks to accommodate the amount of available core memory.
MERGE ERROR	This message is always printed in combination with other messages. Depending on the other message, any of three things may happen after MERGE ERROR is printed: processing may continue, an abort may occur immediately, or an abort may occur after all Merge parameters have been scanned for errors.
MERGE ERROR ILLEGAL DECIMAL DIGIT FILE: n	An input record in file n contains a key field with an illegal decimal digit. This message is always preceded by a one-line printout of the incorrect record, truncated to 131 bytes, if necessary. The incorrect record will be dropped. The range of n is from 0 to 8, where 0 indicates the output file. If the error exceeds the allowed maximum (designated in the Merge parameter control card), the message will print out, preceded by the MERGE ABORT prefix (instead of MERGE ERROR), and the job will abort.
MERGE ERROR INPUT SEQUENCE FILE: n	Input file n is not sequenced within itself. This may occur if a key is outside the boundary of an actual record. This message is always preceded by a one-line printout of the incorrect record, truncated to 131 bytes, if necessary. The incorrect record will be dropped. The range of n is from 0 to 8, where 0 indicates the output file. If the error exceeds the allowed maximum (designated in the Merge parameter control card), the message will print out, preceded by the MERGE ABORT prefix (instead of MERGE ERROR), and the job will abort.
MERGE ERROR INPUT/READ FILE: n	An irrecoverable error has occurred while reading in file n. A blocked variable-length record different from the specified record length will cause this error. This message is always preceded by a one-line printout of the incorrect record, truncated to 131 bytes, if necessary. The incorrect record will be dropped. The range of n is from 0 to 8, where 0 indicates the output file. If the error exceeds the allowed maximum (designated in the Merge parameter control card), the message will print out, preceded by the MERGE ABORT prefix (instead of MERGE ERROR), and the job will abort.
MERGE ERROR OPENING FILE: n	File n could not be properly opened because of an incorrect or missing ASSIGN, because of I/O device failure, or because of improper input header specification causing the entire file to be bypassed. The job will abort after all the remaining input files have been opened.

Table 4. Merge Error and Information Messages (cont.)

Message	Comments
MERGE ERROR SPECIFICATION CARD FIELD	This error message indicates that an error has been encountered in one of the Merge parameter control cards. Any of several messages follow this one to indicate which card is in error: IHDR FIELD NOT BLANK, F, OR 1-9 – the IHDR parameter of the Merge FILE control card contains an invalid digit (see Chapter 6). INPUT BLOCKING – the input blocking factor of the Merge BLOCK card is invalid (see Chapter 6). LOGICAL RECORD LENGTH – the input specification of the Merge REC card is invalid (see Chapter 6). OUTPUT RECORD LENGTH – the output specification of the Merge REC card is invalid (see Chapter 6).
MERGE FILE: n,RECORDS: r	This message is output ten times at the conclusion of a Merge task to indicate the number of logical records (r) processed for each file. The range of n is from 0 to 8, where 0 indicates the output file, and 1-8 represent the input file DCBs, even if not used. For the last iteration of the message, n is equal to the total number of records output and r specifies the total number of input records from all files. In most cases, the total input should equal the amount output.
MERGE SPECIFICATIONS	This precedes a list of all specification control record images.
MERGE SUCCESSFULLY COMPLETED	This is output at the end of the Merge task if no abort conditions have been detected. This message indicates that other tasks within the job will be executed.
NUMBER OF INPUT FILES	The number of input files specified is invalid. Specifically, the FILES parameter of the Merge FILE card contains other than a value from 1 to 8. (See the Merge FILE card description in Chapter 6.) This error will cause a job abort after all remaining Merge parameter control cards have been scanned for errors.
NUMBER OF MERGE KEYS	The specification for number of merge keys is invalid. Specifically, the Merge KEYS card either contains an illegal character or specifies a number that is outside the legal range of values.† (See the KEYS card description in Chapter 6.) This error will cause a job abort after all remaining Merge parameter control cards have been scanned for errors.
OUT OF SEQUENCE DROPS	The specification for number of records to be dropped on an out-of-sequence condition is invalid.

†If alphabetic characters are input where numbers are expected, their zones will be dropped and the least significant part of the character will be treated as a number.

Table 4. Merge Error and Information Messages (cont.)

Message	Comments
OUTPUT BLOCKING	The output blocking specification is invalid. Specifically, the output blocking factor on the Merge BLOCK card either contains an illegal character or specifies a number outside the range of legal values (the number can consist of up to three digits).† (See the BLOCK card description in Chapter 6.) This error will cause a job abort after all remaining Merge parameter control cards have been scanned for errors.
OUTPUT RECORD LENGTH	See MERGE ERROR SPECIFICATION CARD FIELD, above.
TRANSFER FILE: n	A header or trailer of the specified file (i.e., file n) could not be handled. This error causes an immediate abort.
TRANSLATION TABLE LOCATION ERROR	The specified start of a user translation table (of collating sequence values) is a value greater than 256; that is, the start parameter of a TRAN card is greater than 256. The replacement record will exceed the standard 256-byte table area and will cause a table overflow.
USER TRANSLATION TABLE	This precedes the list of control records used to form the translation table if input by the user.

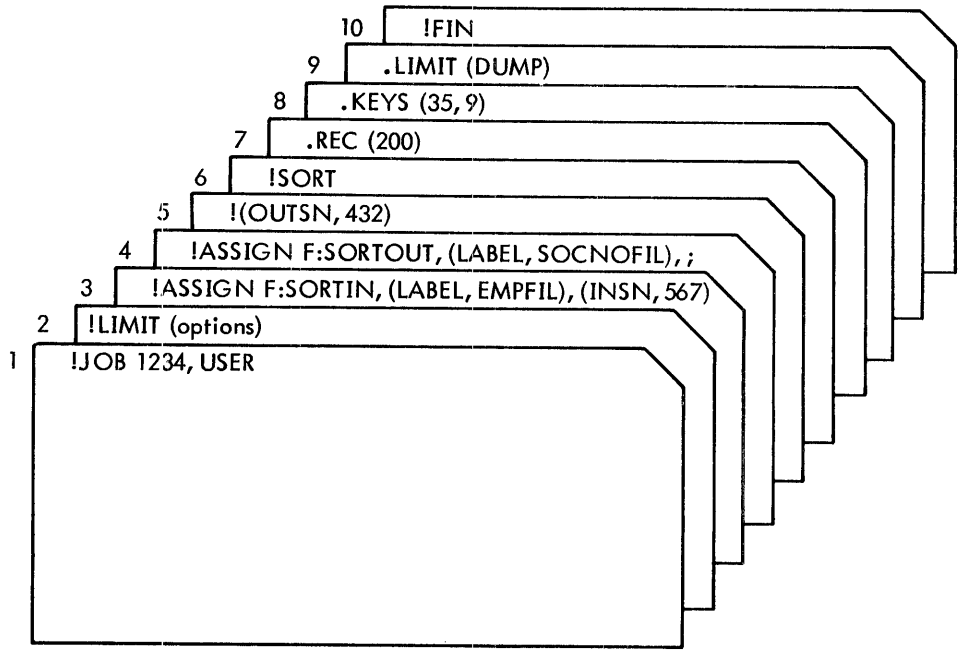
†If alphabetic characters are input where numbers are expected, their zones will be dropped and the least significant part of the character will be treated as a number.

APPENDIX A. SORT OPERATING EXAMPLES

Examples A-1 through A-6 illustrate the card deck setups for different kinds of sort operations:

- Example A-1 produces a sorted file, using the standard sort.
- Example A-2 produces a standard Sort call, output on the RAD from a user-formatted tape, and illustrates multiple key fields and character collating sequence translation.
- Example A-3 defines a special purpose sort in which the user adds own-code and permanently saves the program.
- Examples A-4 through A-6 illustrate Sort use of intermediate storage: Example A-4 is a sequential sort for tape and/or nonrandom disk with four intermediate DCB work files, Example A-5 is a sequential sort for disk with six private scratch packs, and Example A-6 is a random sort for disk with seven private scratch packs. See Chapter 4 for a detailed discussion of Sort use of intermediate storage.

Example A-1. Standard Sort, Output on Tape

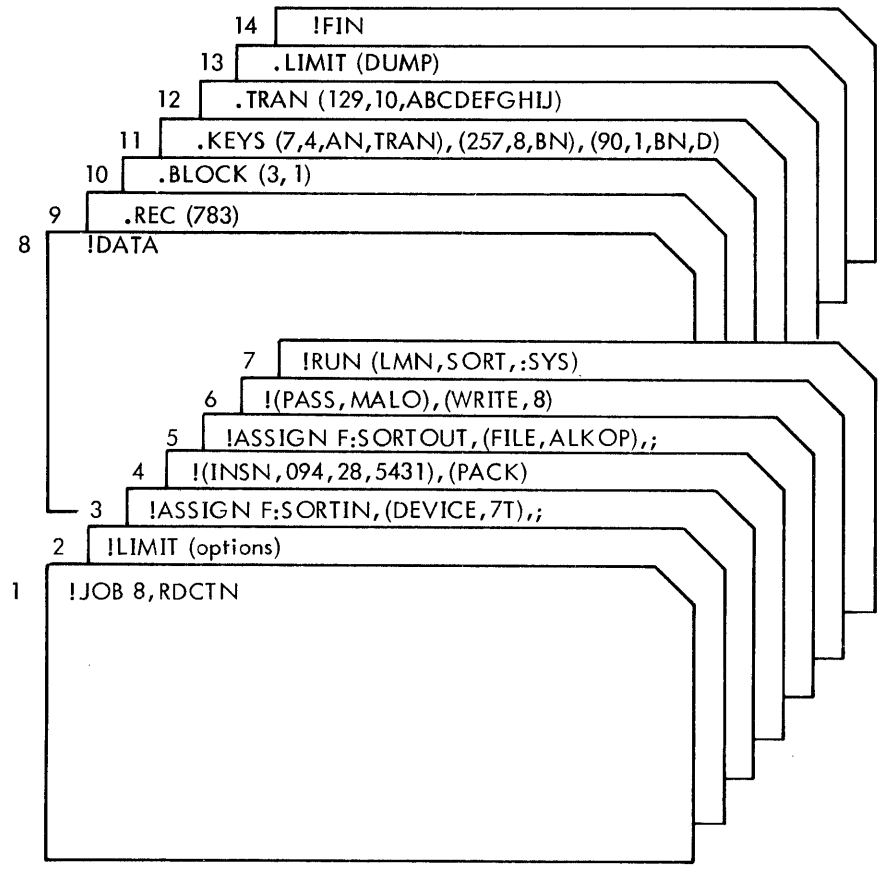


Card	Parameter	Description
1	!JOB	Signals the beginning of a job.
	1234,	Account number. Unless otherwise specified, all data files used during this job will be assumed to belong to this account number.
	USER	Identifies the user.
2	!LIMIT(options)	Control command that specifies the maximum values for various system resources required by the job. The user should supply the options appropriate to each job. For these options, see the BPM/BP,RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.
3	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SORTIN,	Specifies the symbolic name of Sort's input DCB.

Example A-1. Standard Sort, Output on Tape (cont.)

Card	Parameter	Description
	(LABEL,EMPFIL),	Specifies that the input file is a monitor-formatted tape file named EMPFIL.
	(INSN,567)	Specifies that the input file is contained on reel number 567.
4	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SORTOUT,	Specifies the symbolic name of Sort's output DCB.
	(LABEL,SOCNOFIL),	Specifies that the output file is to be a monitor-formatted tape file named SOCNOFIL.
	;	Signals that card 5 is a continuation of card 4.
5	!(OUTSN,432)	Specifies that the output file is to be written on reel number 432.
6	!SORT	Calls the Sort processor from the system library.
7	.REC (200)	Specifies that the input file consists of 200-byte logical records.
8	.KEYS (35,9)	Specifies that the input records are to be sorted on a nine-byte key field that starts in byte 35 of each logical record. By default, the key is alphanumeric and the sort is in ascending sequence.
9	.LIMIT (DUMP)	Requests that Sort and its overlays be dumped if an error occurs.
10	!FIN	Signals the end of the job.

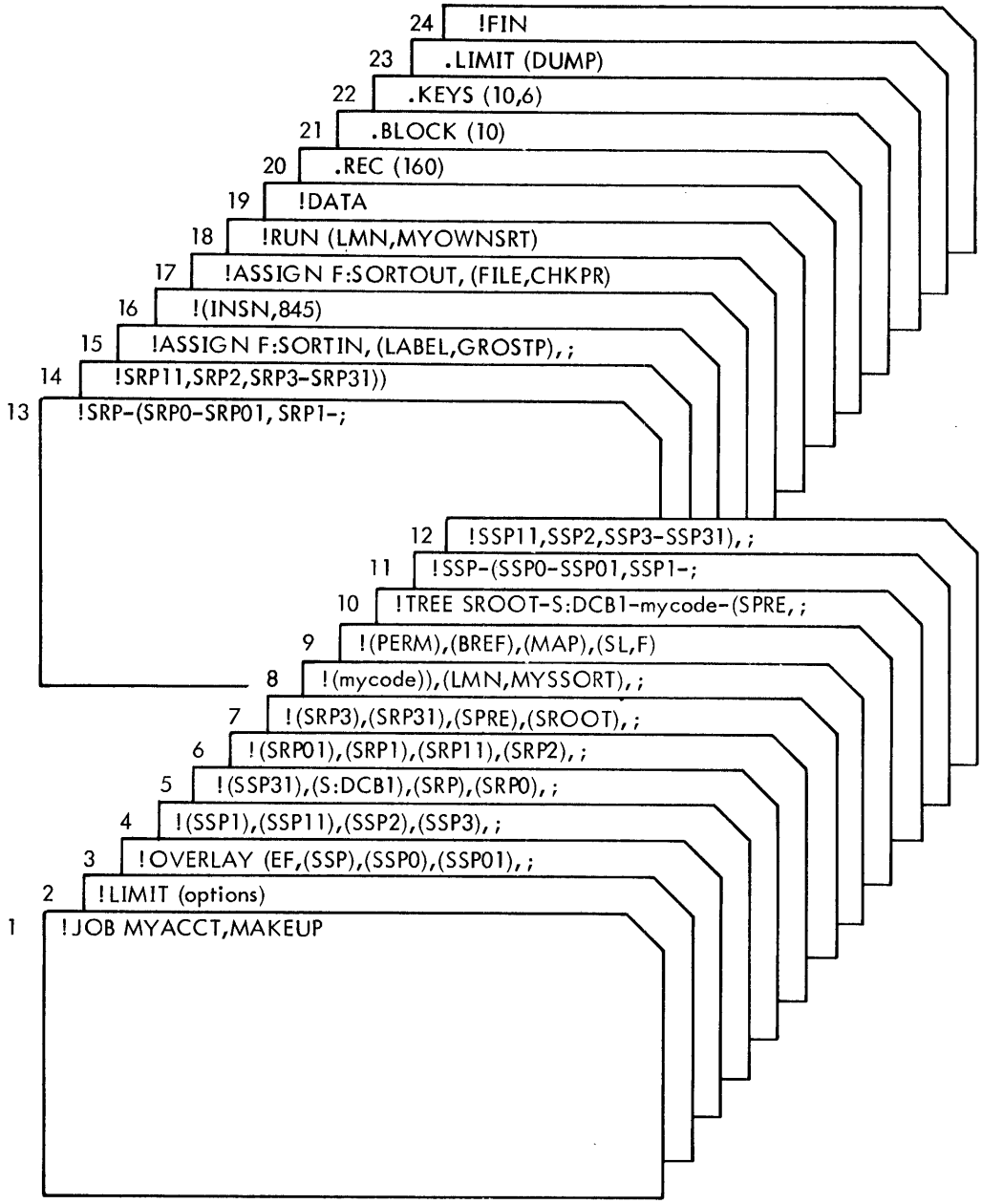
Example A-2. Standard Sort Call, Output on RAD from User-Formatted Tapes



Example A-2. Standard Sort Call, Output on RAD from User-Formatted Tapes (cont.)

Card	Parameter	Description
1	!JOB	Signals the beginning of a job.
	8,	Account number. Unless otherwise specified, all data files used during this job will be assumed to belong to this account number.
	RDCTN	Identifies the user.
2	!LIMIT (options)	Control command that specifies the maximum values for various system resources required by the job. The user should supply the options appropriate to each job. For these options, see the BPM/BP,RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.
3	!ASSIGN	A control command that conveys file characteristics to a Sort's DCBs.
	F:SORTIN,	Specifies the symbolic name of Sort's input DCB.
	(DEVICE,7T),	Specifies that the input file is a 7-track tape.
	;	Signals that card 4 is a continuation of card 3.
4	!(INSN,094,28,5431),	Specifies that the input file is contained on reels 094, 28, and 5431.
	(PACK)	Specifies that the input file is written in packed binary mode.
5	!ASSIGN	A control command that conveys file characteristics to a Sort's DCBs.
	F:SORTOUT,	Specifies the symbolic name of Sort's output DCB.
	(FILE,ALKOP),	Specifies that the sorted output is to be placed on the RAD under the name ALKOP.
	;	Signals that card 6 is a continuation of card 5.
6	!(PASS,MALO),	Assigns the password MALO to the file. This password must be specified by any future user of the file.
	(WRITE,8)	Specifies that only jobs run under account number 8 may write into the output file.
7	!RUN	A control command that specifies a designated program is to be executed. This command may be used in lieu of a SORT control command.
	(LMN,SORT,:SYS)	Designates the load module (processor) named SORT, in the system library, as the program to be executed.
8	!DATA	Signals that a data deck follows. This command must be used when RUN is used to initiate a sort.
9	.REC (783)	Specifies that the input file consists of 783-byte logical records.
10	.BLOCK (3,1)	Specifies a blocking factor of 3 for the input file and 1 for the output file.
11	.KEYS	Specifies that the input file is to be sorted on certain key fields (three key fields in this case), and describes these key fields.
	(7,4,AN,TRAN),	Specifies that key 1 is a four-byte alphanumeric key that starts in byte 7 of each logical record and that the key is to be translated.
	(257,8,BN),	Specifies that key 2 is an eight-byte binary key that starts in byte 257 of each logical record. By default, the sort is in ascending sequence.
	(90,1,BN,D)	Specifies that key 3 is a one-byte binary key that starts in byte 90 of each logical record. The sort is in descending sequence.
12	.TRAN (129,10, ABCDEFGHIJ)	Contains the user-supplied set of character collating sequence values for translation.
13	.LIMIT (DUMP)	Requests that Sort and its overlays be dumped if an error occurs.
14	!FIN	Signals the end of the job.

Example A-3. Sort Jobs Required to Implement User Own-Code



Note: The job loads the own-code to the Sort processor, forming MYSSORT. If the own-code modules are files, the names must also be included in the OVERLAY command and in the root of the TREE command. For example, if they are all in, say, the SORTLIB account, then that account must be added to each module name (e.g., SSP would become SSP,SORTLIB).

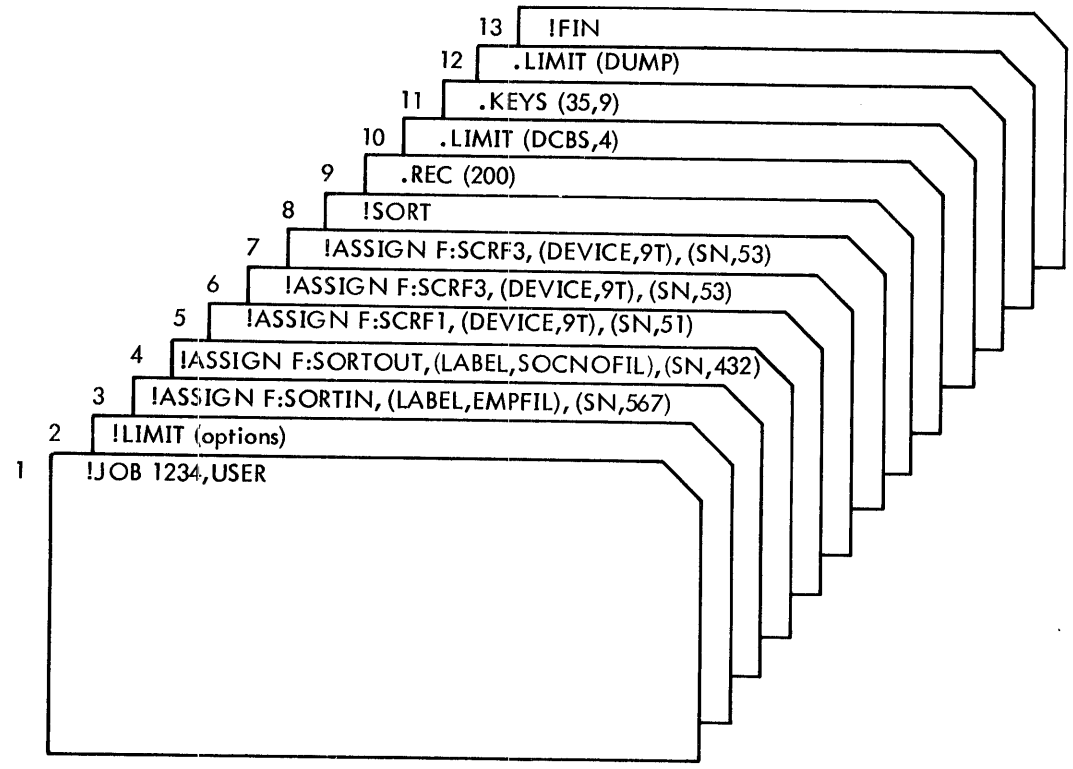
Example A-3. Sort Jobs Required to Implement User Own-Code (cont.)

Card	Parameter	Description
1	!JOB	Signals the beginning of the job.
	MYACCT,	Specifies the account number.
	MAKEUP	Identifies the user.
2	!LIMIT (options)	Control command that specifies the maximum values for various system resources required by the job. For these options, see the BPM/BP,RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.
3	!OVERLAY	A control command that directs the loader to form an executable program from various component programs. This command must be followed by a TREE command.
	(EF,(SSP), (SSP0),(SSP01),	Specifies the modules SSP, SSP0, and SSP01.
	;	Signals that card 4 is a continuation of card 3.
4	(SSP1),(SSP11), (SSP2),(SSP3),	Specifies the modules SSP1, SSP11, SSP2, and SSP3.
	;	Signals that card 5 is a continuation of card 4.
5	(SSP31),(S:DCB1), (SRP),(SRP0),	Specifies the modules SSP31, S:DCB1, SRP, and SRP0.
	;	Signals that card 6 is a continuation of card 5.
6	(SRP01),(SRP1), (SRP11),(SRP2),	Specifies modules SRP01, SRP1, SRP11, and SRP2.
	;	Signals that card 7 is a continuation of card 6.
7	(SRP3),(SRP31), (SPRE),(SROOT),	Specifies modules SRP3, SRP31, SPRE, and SROOT.
	;	Signals that card 8 is a continuation of card 7.
8	(mycode), (LMN,MYSSORT),	Specifies the module (mycode) containing the user's program and specifies that load module MYSSORT is to be formed – this is the special sort with appended own-code.
	;	Signals that card 9 is a continuation of card 8.
9	(PERM),(BREF), (MAP),(SL,F)	Specifies that the load module is to be saved as a permanent file, the branch referencing loading mode is to be used, a load map is to be listed, and an error severity level of F is to be tolerated in the load module.

Example A-3. Sort Jobs Required to Implement User Own-Code (cont.)

Card	Parameter	Description
10	!TREE	Defines structure of MYSSORT.
	SROOT-S:DCB1- mycode-(SPRE,	Specifies root segments SROOT and S:DCB1, the user's own code, and overlay segment SPRE.
	;	Signals that card 11 is a continuation of card 10.
11	!SSP-(SSP0- SSP01,SSP1-	Specifies overlay segments SSP, SSP0, SSP01, and SSP1.
	;	Signals that card 12 is a continuation of card 11.
12	!SSP11,SSP2, SSP3-SSP31),	Specifies overlay segments SSP11, SSP2, SSP3, and SSP31.
	;	Signals that card 13 is a continuation of card 12.
13	!SRP-(SRP0- SRP01,SRP1-	Specifies overlay segments SRP, SRP0, SRP01, and SRP1.
	;	Signals that card 14 is a continuation of card 13.
14	!SRP11,SRP2, SRP3-SRP31))	Specifies overlay segments SRP11, SRP2, SRP3, and SRP31.
15	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SORTIN,	Specifies the symbolic name of Sort's input DCB.
	(LABEL,GROSTP),	Specifies that the input file is a monitor-formatted tape file name GROSTP.
	;	Signals that card 16 is a continuation of card 15.
16	!(INSN,845)	Specifies that the input file is on reel 845.
17	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SORTOUT,	Specifies the symbolic name of Sort's output DCB.
	(FILE,CHKPR)	Specifies that the sorted output is to be placed on the RAD under the name CHKPR.
18	!RUN	A control command that specifies a designated program is to be executed.
	(LMN,MYOWNSRT)	Designates the program formed under the specified name, under the account number of the current job.
19	!DATA	Signals the monitor that a data deck follows. This command must be used when !RUN is used to initiate a sort.
20	.REC (160)	Specifies that the input file is an input file of 160-byte logical records.
21	.BLOCK (10)	Specifies that the input and output files are to have a blocking factor of 10.
22	.KEYS (10,6)	Specifies that the input records are to be sorted on a six-byte key field that starts in byte 10 of each logical record. By default, the key is alphanumeric and the sort is in ascending sequence.
23	.LIMIT (DUMP)	Requests that Sort and its overlays be dumped if an error occurs.
24	!FIN	Signals the end of the job.

Example A-4. Sequential Sort for Tape and/or Disk, with Four Intermediate DCBs



Card	Parameter	Description
1	!JOB	Signals the beginning of the job.
	1234,	Account number. Unless otherwise specified, all data files used during this job will be assumed to belong to this account number.
	USER	Identifies the user.
2	!LIMIT (options)	Control command that specifies the maximum values for various system resources required by the job. The user should supply the options appropriate to each job. For these options, see the BPM/BP,RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.
3	!ASSIGN	A control command that conveys file characteristics to Sort's DCBs.
	F:SORTIN,	Specifies the symbolic name of Sort's input DCB.
	(LABEL,EMPFIL),	Specifies that the input file is a monitor-formatted tape file named EMPFIL.
	(SN,567)	Specifies that the input file is contained on reel number 567.
4	!ASSIGN	A control command that conveys file characteristics to Sort's DCBs.
	F:SORTOUT,	Specifies the symbolic name of Sort's output DCB.
	(LABEL,SOCNOFIL),	Specifies that the output file is to be a monitor-formatted tape file named SOCNOFIL.
	(SN,432)	Specifies that the output file is to be written on reel number 432.

Example A-4. Sequential Sort for Tape and/or Disk, with Four Intermediate DCBs (cont.)

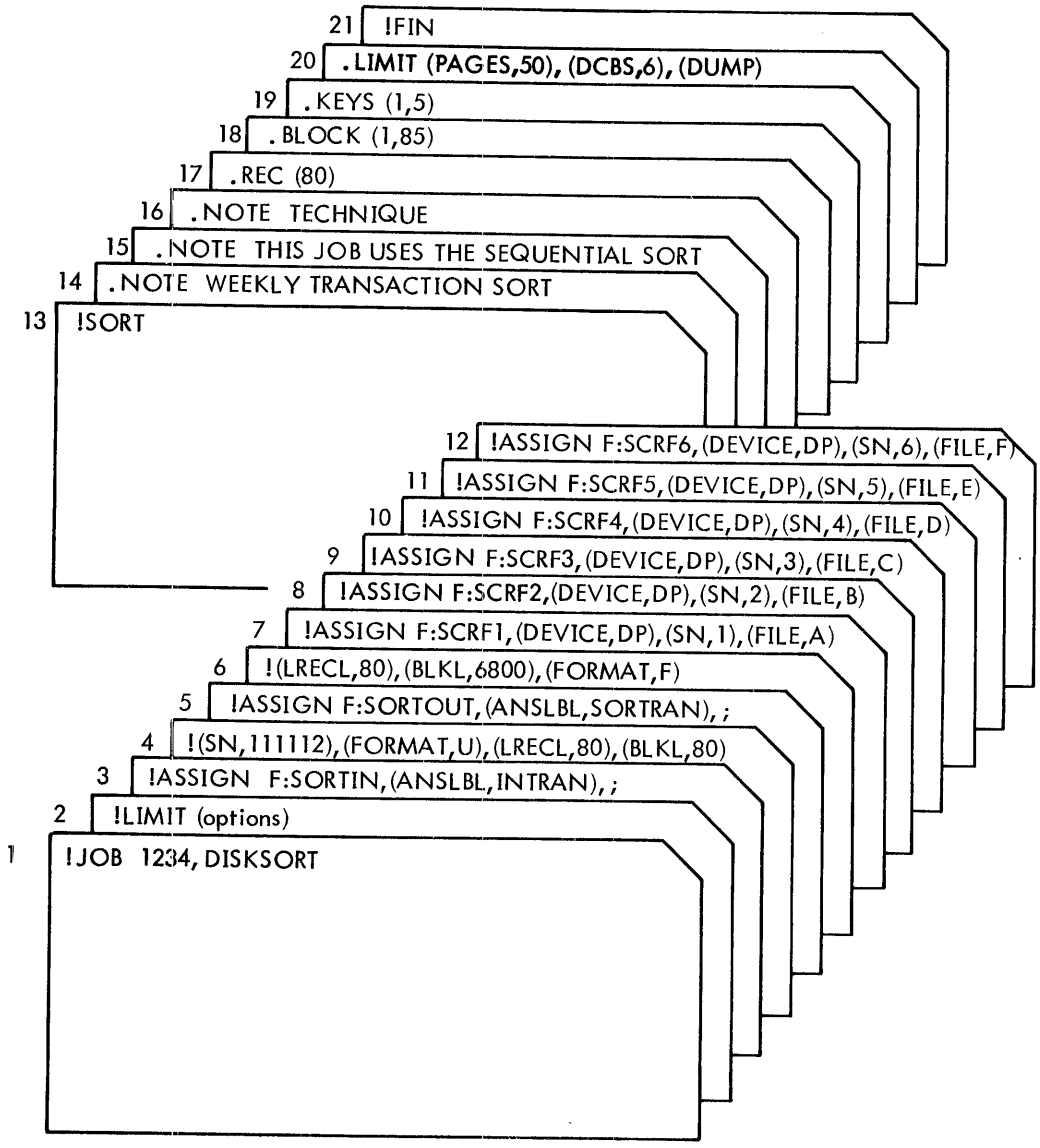
Card	Parameter	Description
5	!ASSIGN	A control command that conveys file characteristics to Sort's DCBs.
	F:SCRF1,	Specifies the symbolic name of the DCB for Sort's first intermediate file.
	(DEVICE,9T),	Specifies that the first intermediate file is to be on a user-formatted 9-track magnetic tape.
	(SN,51)	Specifies the serial number of the tape reel that is to be used for the intermediate file.
6	!ASSIGN	A control command that conveys file characteristics to Sort's DCBs.
	F:SCRF2,	Specifies the symbolic name of the DCB for Sort's second intermediate file.
	(DEVICE,9T),	Specifies that the second intermediate file is to be on a user-formatted 9-track magnetic tape.
	(SN,52)	Specifies the serial number of the tape reel that is to be used for the intermediate file.
7	!ASSIGN	A control command that conveys file characteristics to Sort's DCBs.
	F:SCRF3,	Specifies the symbolic name of the DCB for Sort's third intermediate file.
	(DEVICE,9T)	Specifies that the third intermediate file is to be on a user-formatted 9-track magnetic tape.
	(SN,53)	Specifies the serial number of the tape reel that is to be used for the intermediate file.
8	!SORT	Calls the Sort processor from the system library.
9	.REC (200)	Specifies that the input file consists of 200-byte logical records.
10	.LIMIT (DCBS,4)	Specifies that the Sort program is to use four intermediate work files.
11	.KEYS (35,9)	Specifies that the input records are to be sorted on a nine-byte key field that starts in byte 35 of each logical record. By default, the key is alphanumeric and the sort is in ascending sequence.
12	.LIMIT (DUMP)	Requests that Sort and its overlays be dumped if an error occurs.
13	!FIN	Signals the end of the job.

Notes:

1. When assigning intermediate files to magnetic tape, the assignment should always be to a user-formatted tape (i.e., DEVICE). If the assignment is to a monitor-formatted file (i.e., LABEL), the sort will not be as efficient.

2. If the DCBS parameter of the .LIMIT card (see card 10) did not specify the number of tapes, Sort would use six intermediate files. The first three files would be on 9-track magnetic tape, and the last three would be on public storage.

Example A-5. Sequential Sort for Disk, with Six Private Scratch Packs



Card	Parameter	Description
1	!JOB	Signals the beginning of the job.
	1234,	Account number. Unless otherwise specified, all data files used during this job will be assumed to belong to this account number.
	DISKSORT	Identifies the user.
2	!LIMIT (options)	Control command that specifies the maximum values for various system resources required by the job. The user should supply the options appropriate to each job. For these options, see the BPM/BP, RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.

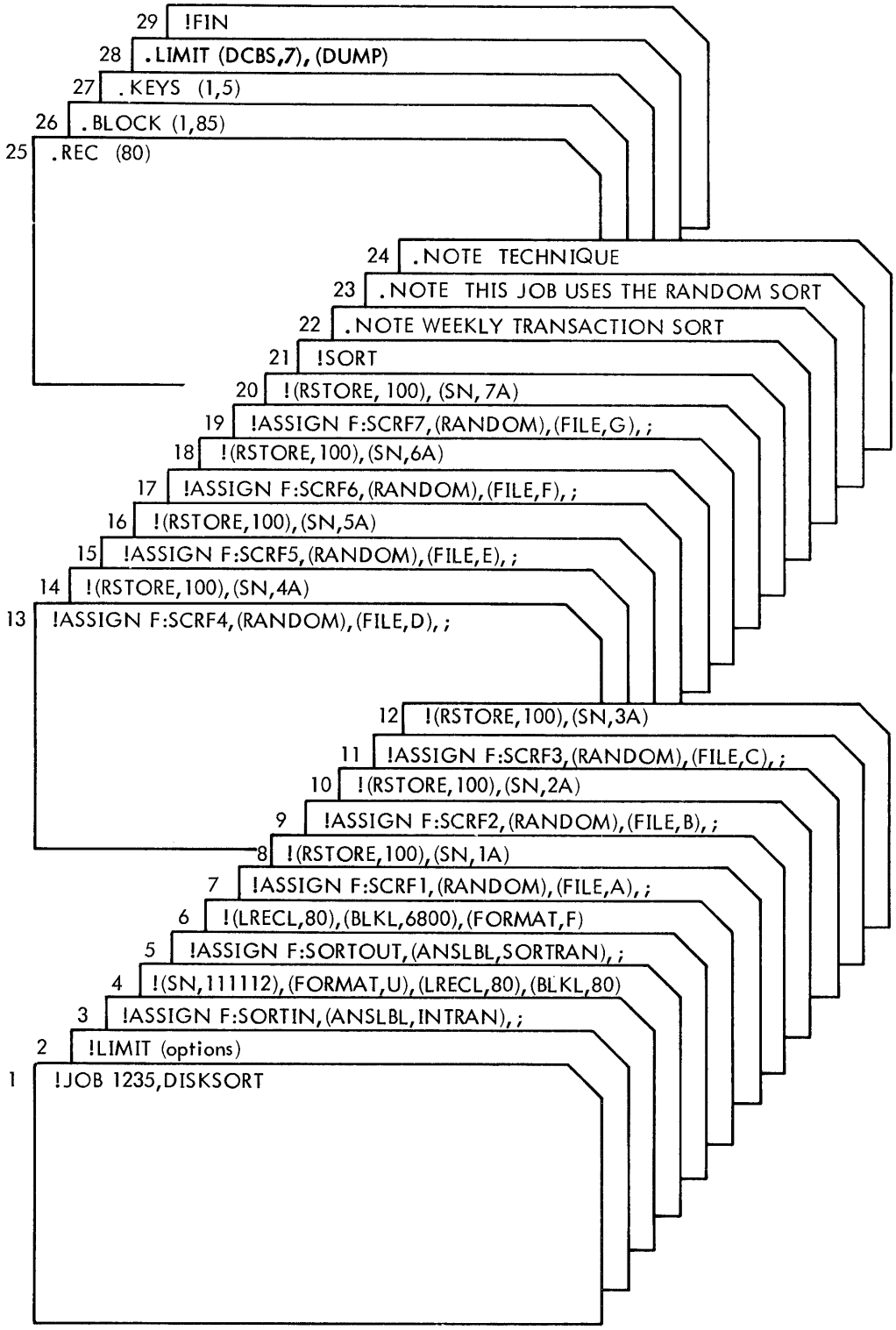
Example A-5. Sequential Sort for Disk, with Six Private Scratch Packs (cont.)

Card	Parameter	Description
3	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SORTIN,	Specifies the symbolic name of Sort's input DCB.
	(ANSLBL,INTRAN),	Indicates that the input file is an ANSI formatted tape with the label INTRAN.
	;	Signals that card 4 is a continuation of card 3.
4	!(SN,111112),	Specifies the serial number of the tape reel that is to be used for file input.
	(FORMAT,U),	Specifies that the format of the physical records on the input file is undefined format. In other words, each record is a block. If all input records are the same length, ANSI fixed format (F) can be specified for the output file. If the input records are not all the same length, however, the output file should be specified as ANSI variable format (V).
	(LRECL,80),	Specifies that the input record length is 80 bytes.
	(BLKL,80)	Specifies that the input block length is 80 bytes.
5	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SORTOUT,	Specifies the symbolic name of Sort's output DCB.
	(ANSLBL,SORTRAN),	Indicates that the output file is an ANSI formatted tape with the label SORTRAN.
	;	Signals that card 6 is a continuation of card 5.
6	!(LRECL,80),	Specifies that the output record length is 80 bytes.
	(BLKL,6800),	Specifies that the output block length is 6800 bytes.
	(FORMAT,F)	Specifies that the format of the physical records on the output file is fixed format.
7	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF1,	Specifies the symbolic name of the first Sort work file.
	(DEVICE,DP),	Indicates that the storage device (for the first Sort work file) is a disk pack.
	(SN,1),	Indicates that the disk pack is a private disk pack and that its serial number is 1.
	(FILE,A)	Specifies that A is the name of the disk file that is to be assigned to the DCB.
8	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF2,	Specifies the symbolic name of the second Sort work file.
	(DEVICE,DP),	Indicates that the storage device (for the second Sort work file) is a disk pack.
	(SN,2),	Indicates that the disk pack is a private disk pack and that its serial number is 2.
	(FILE,B)	Specifies that B is the name of the disk file that is to be assigned to the DCB.
9	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF3,	Specifies the symbolic name of the third Sort work file.

Example A-5. Sequential Sort for Disk, with Six Private Scratch Packs (cont.)

Card	Parameter	Description
	(DEVICE,DP),	Indicates that the storage device (for the third Sort work file) is a disk pack.
	(SN,3),	Indicates that the disk pack is a private disk pack and that its serial number is 3.
	(FILE,C)	Specifies that C is the name of the disk file that is to be assigned to the DCB.
10	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF4,	Specifies the symbolic name of the fourth Sort work file.
	(DEVICE,DP),	Indicates that the storage device (for the fourth Sort work file) is a disk pack.
	(SN,4),	Indicates that the disk pack is a private disk pack and that its serial number is 4.
	(FILE,D)	Specifies that D is the name of the disk file that is to be assigned to the DCB.
11	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF5,	Specifies the symbolic name of the fifth Sort work file.
	(DEVICE,DP),	Indicates that the storage device (for the fifth Sort work file) is a disk pack.
	(SN,5),	Indicates that the disk pack is a private disk pack and that its serial number is 5.
	(FILE,E)	Specifies that E is the name of the disk file that is to be assigned to the DCB.
12	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF6,	Specifies the symbolic name of the sixth Sort work file.
	(DEVICE,DP),	Indicates that the storage device (for the sixth Sort work file) is a disk pack.
	(SN,6),	Indicates that the disk pack is a private disk pack and that its serial number is 6.
	(FILE,F)	Specifies that F is the name of the disk file that is to be assigned to the DCB.
13	!SORT	Calls the Sort processor from the system library.
14	.NOTE	Indicates commentary that this is a weekly transaction sort.
15,16	.NOTE	Indicates commentary that this job uses the sequential sort technique.
17	.REC (80)	Specifies that the input file consists of 80-byte logical records.
18	.BLOCK (1,85)	Specifies that the input file blocking factor is 1 and the output file blocking factor is 85.
19	.KEYS (1,5)	Specifies that the input records are to be sorted on a five-byte key field that starts in byte 1 of each logical record. By default, the sort key is alphanumeric and the sort is in ascending sequence.
20	.LIMIT (PAGES,50),	Specifies that the Sort program is to occupy no more than 50 pages of memory (a page = 512 words).
	(DCBS,6),	Specifies that the Sort program is to use six intermediate work files.
	(DUMP)	Requests that Sort and its overlays be dumped if an error occurs.
21	!FIN	Signals the end of the job.

Example A-6. Random Sort for Disk, with Seven Private Scratch Packs



Example A-6. Random Sort for Disk, with Seven Private Scratch Packs (cont.)

Card	Parameter	Description
1	!JOB	Signals the beginning of the job.
	1235,	Account number. Unless otherwise specified, all data files used during this job will be assumed to belong to this account number.
	DISKSORT	Identifies the user.
2	!LIMIT (options)	Control command that specifies the maximum values for various system resources required by the job. The user should supply the options appropriate to each job. For these options, see the BPM/BP,RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.
3	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SORTIN,	Specifies the symbolic name of Sort's input DCB.
	(ANSLBL,INTRAN),	Indicates that the input file is an ANSI formatted tape with the label INTRAN.
	;	Signals that card 4 is a continuation of card 3.
4	!(SN,111112),	Specifies the serial number of the tape reel that is to be used for file input.
	(FORMAT,U),	Specifies that the format of the physical records on the input file is undefined format. In other words, each record is a block. If all input records are the same length, ANSI fixed format (F) can be specified for the output file. If the input records are not all the same length, however, the output file should be specified as ANSI variable format (V).
	(LRECL,80),	Specifies that the input record length is 80 bytes.
	(BLKL,80)	Specifies that the input block length is 80 bytes.
5	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SORTOUT,	Specifies the symbolic name of Sort's output DCB.
	(ANSLBL,SORTRAN),	Indicates that the output file is an ANSI formatted tape with the label SORTRAN.
	;	Signals that card 6 is a continuation of card 5.
6	!(LRECL,80),	Specifies that the output record length is 80 bytes.
	(BLKL,6800),	Specifies that the output block length is 6800 bytes.
	(FORMAT,F)	Specifies that the format of the physical records on the output file is fixed format.
7	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF1,	Specifies the symbolic name of the first Sort work file.
	(RANDOM),	Specifies that the data in the file is to be written in contiguous areas of a random access device (in this case a disk pack) and accessed by specifying the starting block number.
	(FILE,A),	Specifies that A is the name of the disk file that is to be assigned to the DCB.
	;	Signals that card 8 is a continuation of card 7.

Example A-6. Random Sort for Disk, with Seven Private Scratch Packs (cont.)

Card	Parameter	Description
8	!(RSTORE,100)	Specifies that 100 granules of disk storage are to be allocated to the RANDOM file.†
	(SN,1A)	Specifies the serial number of the disk pack to be used.
9	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF2,	Specifies the symbolic name of the Sort work file.
	(RANDOM),	Specifies that the data in the file is to be written in contiguous areas of a random access device (in this case a disk pack) and accessed by specifying the starting block number.
	(FILE,B),	Specifies that B is the name of the disk file that is to be assigned to the DCB.
	;	Signals that card 10 is a continuation of card 9.
10	!(RSTORE,100),	Specifies that 100 granules of disk storage are to be allocated to the RANDOM file.†
	(SN,2A)	Specifies the serial number of the disk pack to be used.
11	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF3,	Specifies the symbolic name of the Sort work file.
	(RANDOM),	Specifies that the data in the file is to be written in contiguous areas of a random access device (in this case a disk pack) and accessed by specifying the starting block number.
	(FILE,C),	Specifies that C is the name of the disk file that is to be assigned to the DCB.
	;	Signals that card 12 is a continuation of card 11.
12	!(RSTORE,100),	Specifies that 100 granules of disk storage are to be allocated to the RANDOM file.†
	(SN,3A)	Specifies the serial number of the disk pack to be used.
13	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF4,	Specifies the symbolic name of the Sort work file.
	(RANDOM),	Specifies that the data in the file is to be written in contiguous areas of a random access device (in this case a disk pack) and accessed by specifying the starting block number.
	(FILE,D),	Specifies that D is the name of the disk file that is to be assigned to the DCB.
	;	Signals that card 14 is a continuation of card 13.
14	!(RSTORE,100),	Specifies that 100 granules of disk storage are to be allocated to the RANDOM file.†
	(SN,4A)	Specifies the serial number of the disk pack to be used.

†See Chapter 4 for a discussion of the number of granules needed.

Example A-6. Random Sort for Disk, with Seven Private Scratch Packs (cont.)

Card	Parameter	Description
15	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF5,	Specifies the symbolic name of the Sort work file.
	(RANDOM),	Specifies that the data in the file is to be written in contiguous areas of a random access device (in this case a disk pack) and accessed by specifying the starting block number.
	(FILE,E),	Specifies that E is the name of the disk file that is to be assigned to the DCB.
	;	Signals that card 16 is a continuation of card 15.
16	!(RSTORE,100),	Specifies that 100 granules of disk storage are to be allocated to the RANDOM file.[t]
	(SN,5A)	Specifies the serial number of the disk pack to be used.
17	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF6,	Specifies the symbolic name of the Sort work file.
	(RANDOM),	Specifies that the data in the file is to be written in contiguous areas of a random access device (in this case a disk pack) and accessed by specifying the starting block number.
	(FILE,F),	Specifies that F is the name of the disk file that is to be assigned to the DCB.
	;	Signals that card 18 is a continuation of card 17.
18	!(RSTORE, 100),	Specifies that 100 granules of disk storage are to be allocated to the RANDOM file.[t]
	(SN,6A)	Specifies the serial number of the disk pack to be used.
19	!ASSIGN	A control command that conveys file characteristics to the Sort's DCBs.
	F:SCRF7,	Specifies the symbolic name of the Sort work file.
	(RANDOM),	Specifies that the data in the file is to be written in contiguous areas of a random access device (in this case a disk pack) and accessed by specifying the starting block number.
	(FILE,G),	Specifies that G is the name of the disk file that is to be assigned to the DCB.
	;	Signals that card 20 is a continuation of card 19.
20	!(RSTORE,100),	Specifies that 100 granules of disk storage are to be allocated to the RANDOM file.[t]
	(SN,7A)	Specifies the serial number of the disk pack to be used.
21	!SORT	Calls the Sort processor from the system library.

[t] See Chapter 4 for a discussion of the number of granules needed.

Example A-6. Random Sort for Disk, with Seven Private Scratch Packs (cont.)

Card	Parameter	Description
22	.NOTE	Indicates commentary that this is a weekly transaction sort.
23,24	.NOTE	Indicates commentary that this job uses the random sort technique.
25	.REC (80)	Specifies that the input file consists of 80-byte logical records.
26	.BLOCK (1,85)	Specifies that the input file blocking factor is 1 and the output file blocking factor is 85.
27	.KEYS (1,5)	Specifies that the input records are to be sorted on a five-byte key field that starts in byte 1 of each logical record. By default, the sort key is alphanumeric and the sort is in ascending sequence.
28	.LIMIT (DCBS,7) ,	Specifies that the Sort program is to use seven intermediate work files.
	(DUMP)	Requests that Sort and its overlays be dumped if an error occurs.
29	!FIN	Signals the end of the job.

Example A-7. On-Line Sort Session (CP-V)

```
!C INTST                                  User requests a copy of the previously created file INTST.
                                          The file consists of 25 single-character records, in random order.
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
1
2
3
4
5
6
7
8
9
0

!SET F:SORTIN /INTST                      CP-V on-line commands, similar to !ASSIGN in batch,
!SET F:SORTOUT /OUTST                     which specify the files INTST and OUTST to be associated
                                          with Sort's DCBs.
!SORT                                     Calls the Sort processor.
SORT VERSION F00: 04/15/75                Sort identifies itself.
.REC (1,1)                                Specifies that the input file and output file consists of
.REC (1,1)                                1-character records.

.KEYS (1,1)                               Specifies that Sort is to be keyed on one character, in
.KEYS (1,1)                               position 1.
```

Example A-7. On-Line Sort Session (CP-V) (cont.)

```
F\
SEQUENTIAL
RECORDS IN TOURNAMENT:          4094
NUMBER OF MERGE BUFFERS:          12
INTERMEDIATE BUFFER SIZE:      12336
RECORDS INPUT:                    25
RECORDS OUTPUT:                   25

!C OUTST
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
R
S
T
U
V
W
X
Y
Z

:
```

ESCAPE and F keys terminate input.
Sort identifies its technique, and prints the statistics of the Sort.

User requests a copy of the newly created file OUTST.

CP-V prompts for next command.

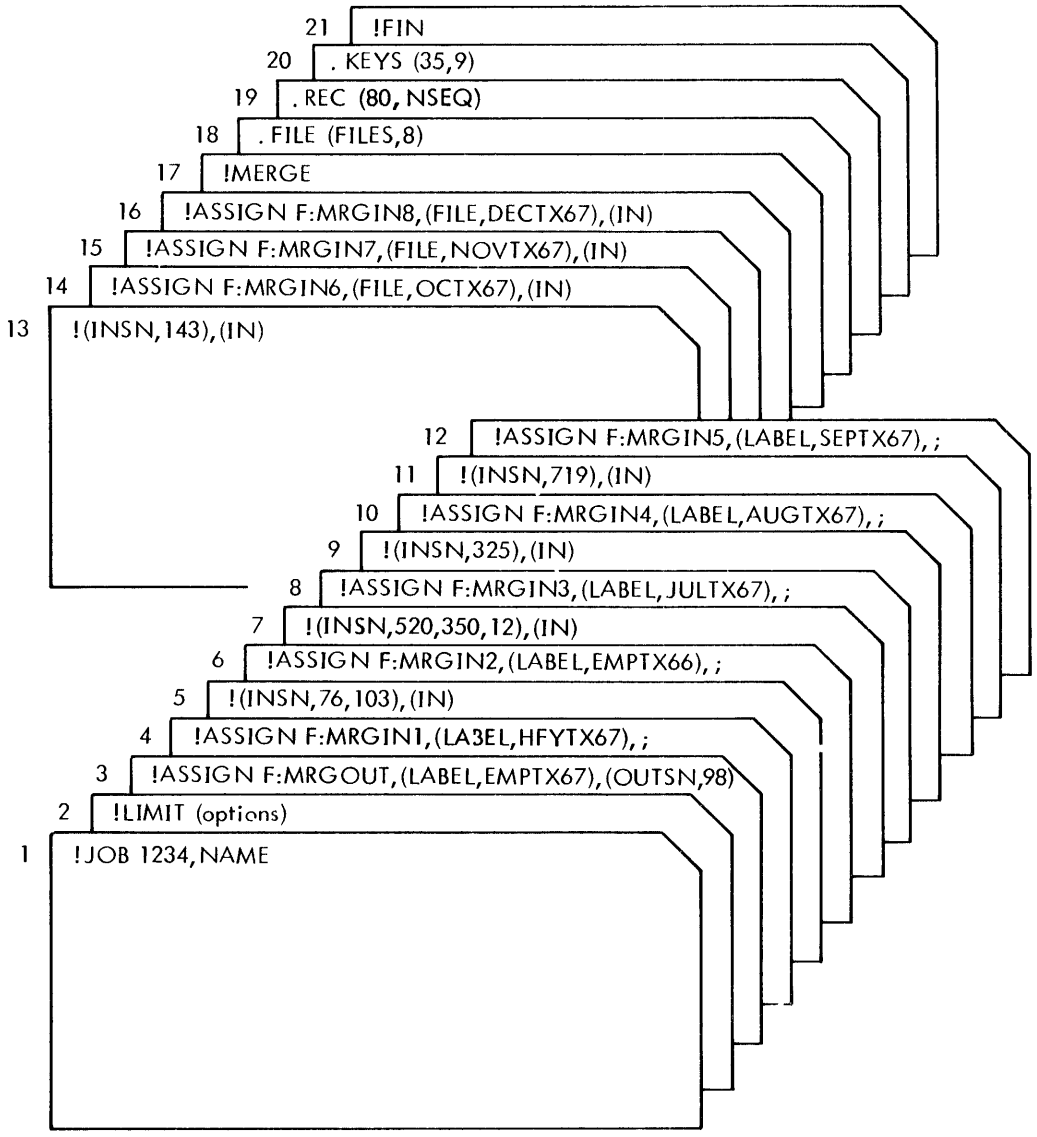
Notes:
1. Underlined characters were sent to the terminal by CP-V; non-underlined characters were typed at the terminal.
2. All terminal input lines were terminated by a carriage return.

APPENDIX B. MERGE OPERATING EXAMPLES

Examples B-1 through B-3 illustrate the card decks for different kinds of merge operations:

- The sample deck setup in Example B-1 produces a merged file from the maximum number of input files. Note that file structures are the same, but that input storage media are mixed (RAD/disk pack and tape).
- The deck setup in Example B-2 produces a standard Merge call, output on public storage from user-formatted tape, and illustrates multiple key fields and character collating sequence translation.
- The sample deck in Example B-3 produces a new program consisting of Merge plus user own-code. The files are identical to those in Example B-2. However, a user's own-code program is linked to the Merge for input header and trailer processing and for input and output own-code processing. The user program is called CHKMRG.

Example B-1. Merged File from Maximum Number of Input Files



Card	Parameter	Description
1	!JOB	Signals the beginning of a job.
	1234,	Account number. Unless otherwise specified, all data files used during the job will be assumed to belong to this account number.

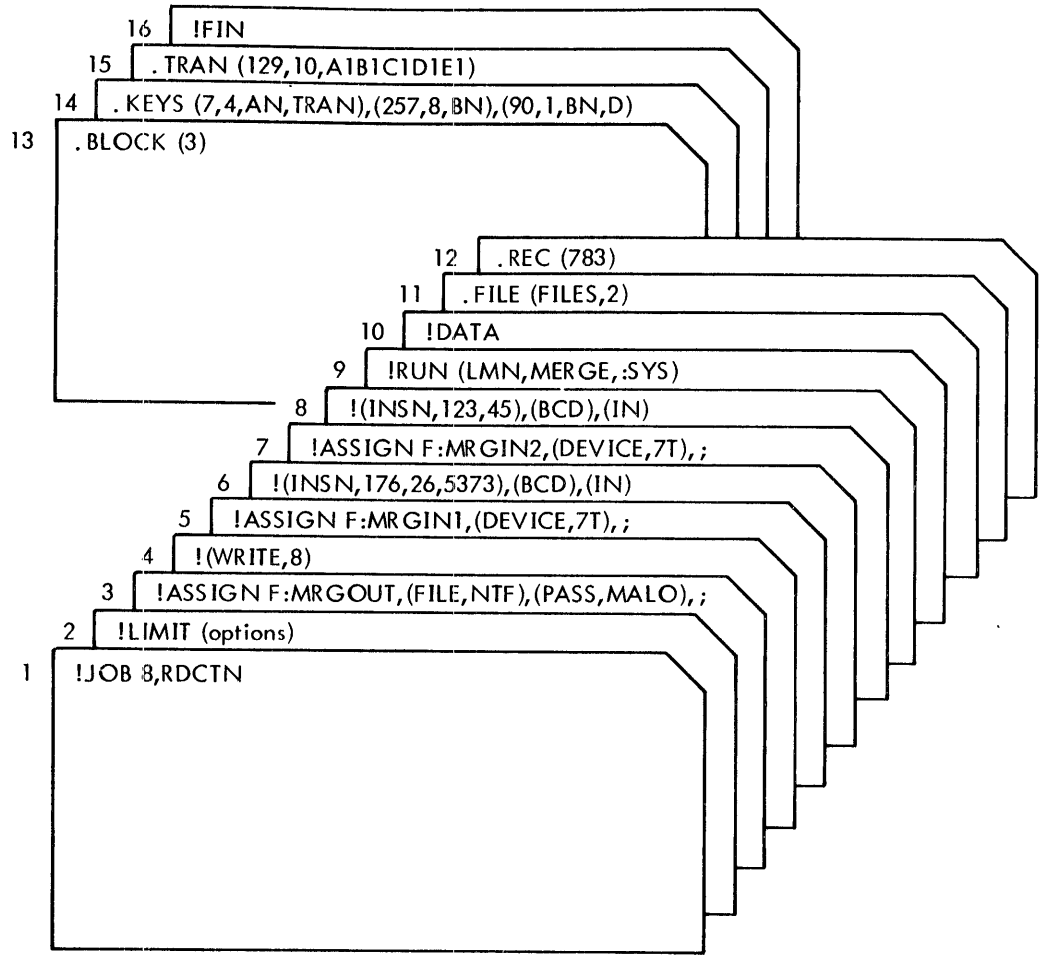
Example B-1. Merged File from Maximum Number of Input Files (cont.)

Card	Parameter	Description
	NAME	Identifies the user.
2	!LIMIT (options)	Control command that specifies the maximum values for various system resources required by the job. The user should supply the options appropriate to each job. For these options, see the BPM/BP, RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.
3	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGOUT,	Specifies the symbolic name of Merge's output DCB.
	(LABEL, EMPTX67),	Specifies that the output file is to be a monitor-formatted tape file named EMPTX67.
	(OUTSN, 98)	Specifies that the output file is to be written on reel number 98.
4	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN1,	Specifies the symbolic name of Merge's input DCB for the first input file.
	(LABEL, HFYTX67),	Specifies that the input file is a monitor-formatted tape file named HFYTX67.
	;	Signals that card 5 is a continuation of card 4.
5	!(INSN, 76, 103),	Specifies that the input file is contained on reels 76 and 103.
	(IN)	Specifies that this is an input file.
6	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN2,	Specifies the symbolic name of Merge's input DCB for the second input file.
	(LABEL, EMPTX66),	Specifies that the input file is to be a monitor-formatted tape file named EMPTX66.
	;	Signals that card 7 is a continuation of card 6.
7	!(INSN, 520, 350, 12),	Specifies that the input file is contained on reels 520, 350, and 12.
	(IN)	Specifies that this is an input file.
8	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN3,	Specifies the symbolic name of Merge's input DCB for the third input file.
	(LABEL, JULTX67),	Specifies that the input file is to be a monitor-formatted tape file named JULTX67.
	;	Signals that card 9 is a continuation of card 8.
9	!(INSN, 325),	Specifies that the input file is contained on reel 325.
	(IN)	Specifies that this is an input file.
10	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN4,	Specifies the symbolic name of Merge's input DCB for the fourth input file.

Example B-1. Merged File from Maximum Number of Input Files (cont.)

Card	Parameter	Description
	(LABEL, AUGTX67),	Specifies that the input file is to be a monitor-formatted tape file named AUGTX67.
	;	Signals that card 11 is a continuation of card 10.
11	!(INSN, 719),	Specifies that the input file is contained on reel 719.
	(IN)	Specifies that this is an input file.
12	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN5,	Specifies the symbolic name of Merge's input DCB for the fifth input file.
	(LABEL, SEPTX67),	Specifies that the input file is to be a monitor-formatted tape file named SEPTX67.
	;	Signals that card 13 is a continuation of card 12.
13	!(INSN, 143),	Specifies that the input file is contained on reel 143.
	(IN)	Specifies that this is an input file.
14	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN6,	Specifies the symbolic name of Merge's input DCB for the sixth input file.
	(FILE, OCTX67),	Specifies that the input file, named OCTX67, is currently on a RAD that may have been loaded because of limited tape drives.
	(IN)	Specifies that this is an input file.
15	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN7,	Specifies the symbolic name of Merge's input DCB for the seventh input file.
	(FILE, NOVTX67),	Specifies that the input file, named NOVTX67, is currently on a RAD that may have been loaded because of limited tape drives.
	(IN)	Specifies that this is an input file.
16	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN8,	Specifies the symbolic name of Merge's input DCB for the last input file.
	(FILE, DECTX67),	Specifies that the input file, named DECTX67, is currently on a RAD that may have been loaded because of limited tape drives.
	(IN)	Specifies that this is an input file.
17	!MERGE	Calls the Merge processor from the system library.
18	.FILE (FILES, 8)	Specifies that eight input files are to be merged.
19	.REC (80, NSEQ)	80 specifies that the input files consist of 80-byte logical records. NSEQ specifies that this job is to ignore out-of-sequence records that may be found in the input files.
20	.KEYS (35, 9)	Specifies that the input files are to be merged according to a nine-byte key field that starts in byte 35 of each logical record. By default, the key is alphanumeric and the merge is in ascending sequence.
21	!FIN	Signals the end of the job.

Example B-2. Standard Merge Call, Output on RAD from User-Formatted Tapes

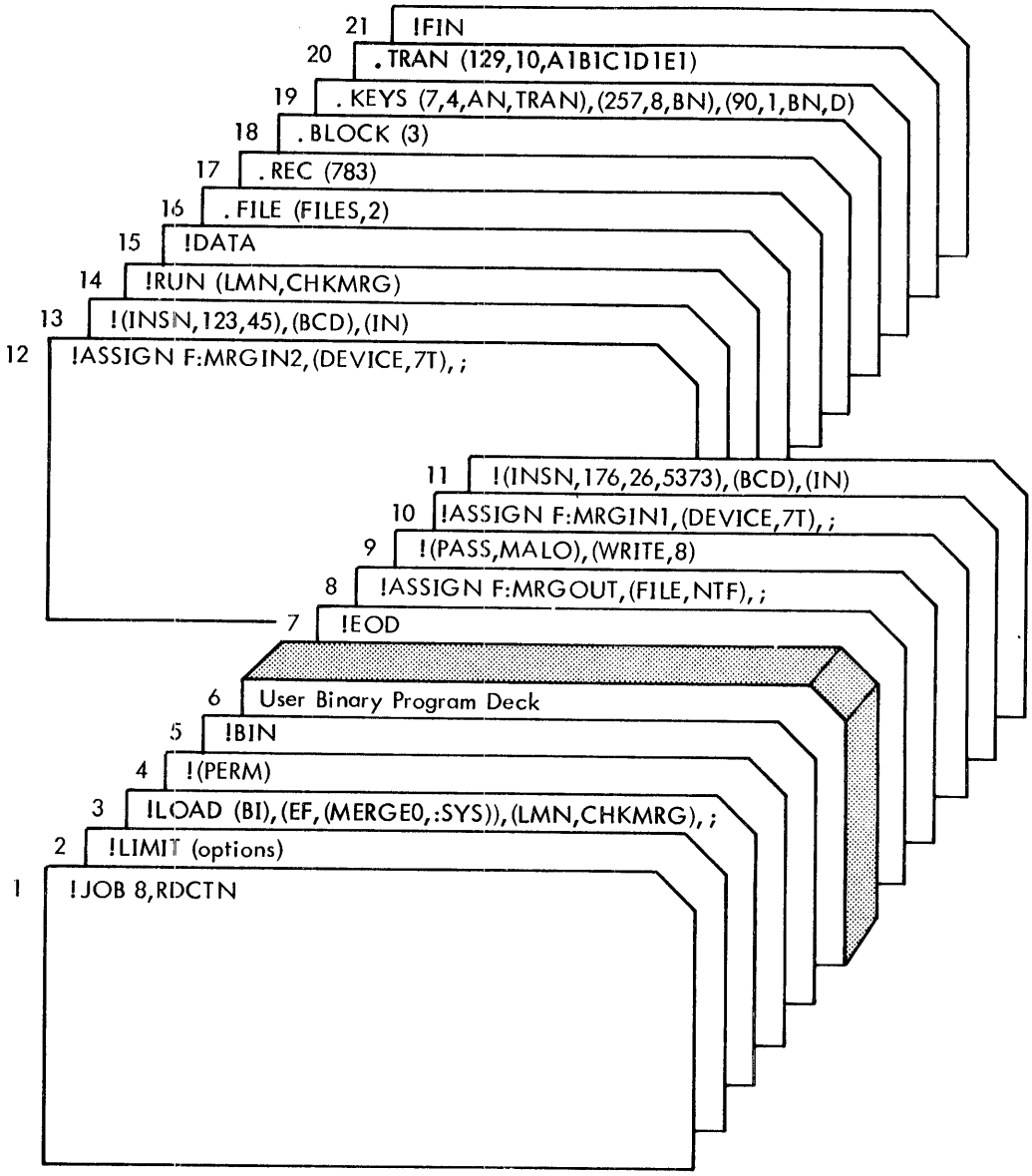


Card	Parameter	Description
1	!JOB	Signals the beginning of a job.
	8,	Account number. Unless otherwise specified, all data files used during this job will be assumed to belong to this account number.
	RDCTN	Identifies the user.
2	!LIMIT (options)	Control command that specifies the maximum values for various system resources required by the job. The user should supply the options appropriate to each job. For these options, see the BPM/BP, RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.
3	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGOUT,	Specifies the symbolic name of Merge's output DCB.
	(FILE, NTF),	Specifies that the sorted output is to be placed on the RAD under the name NTF.
	(PASS, MALO),	Assigns the password MALO to the file. This password must be specified by any future user of the file.
	;	Signals that card 4 is a continuation of card 3.

Example B-2. Standard Merge Call, Output on RAD from User-Formatted Tapes (cont.)

Card	Parameter	Description
4	!(WRITE, 8),	Specifies that only jobs run under account number 8 may write into the output file.
5	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN1,	Specifies the symbolic name of Merge's first input DCB.
	(DEVICE, 7T),	Specifies that the input file is on a 7-track magnetic tape.
	;	Signals that card 6 is a continuation of card 5.
6	!(INSN, 176, 26, 5373),	Specifies that the input file is contained on reels 176, 26, and 5373.
	(BCD),	Specifies that the input file is written in 7-track BCD mode.
	(IN)	Specifies that this is an input file.
7	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN2,	Specifies the symbolic name of Merge's second input DCB.
	(DEVICE, 7T),	Specifies that the input file is on a 7-track magnetic tape.
	;	Signals that card 8 is a continuation of card 7.
8	!(INSN, 123, 45),	Specifies that the input file is contained on reels 123 and 45.
	(BCD),	Specifies that the input file is written in BCD mode.
	(IN)	Specifies that this is an input file.
9	!RUN	A control command that specifies a designated program is to be executed. This command may be used in lieu of a MERGE command.
	(LMN, MERGE, :SYS)	Designates the load module (processor) named MERGE, in the system library, as the program to be executed.
10	!DATA	Signals the monitor that a data deck follows. This command must be used when !RUN is used to initiate a merge.
11	. FILE (FILES, 2)	Specifies that two input files are to be merged.
12	. REC (783)	Specifies that the input files consist of 783-byte logical records.
13	. BLOCK (3)	Specifies that the input file blocking factor is 3. By default, the output file blocking factor is also 3.
14	. KEYS	Specifies that the input files are to be merged according to certain key fields (three key fields in this case).
	(7, 4, AN, TRAN),	Specifies that key 1 is a four-byte alphanumeric key which starts in byte 7 of each logical record and that the key is to be translated.
	(257, 8, BN),	Specifies that key 2 is an eight-byte binary key that starts in byte 257 of each logical record. By default, the merge is in ascending sequence.
	(90, 1, BN, D)	Specifies that key 3 is a one-byte binary key that starts in byte 90 of each logical record. The merge is in descending sequence.
15	. TRAN (129, 10, A1B1 C1D1E1)	Indicates the user-supplied set of character collating sequence values for translation of key 1.
16	!FIN	Signals the end of the job.

Example B-3. Merged File from Merge Program Plus User Own-Code, Output on RAD



Card	Parameter	Description
1	!JOB	Signals the beginning of a job.
	8,	Account number. Unless otherwise specified, all data files used during this job will be assumed to belong to this account number.
	RDCTN	Identifies the user.
2	!LIMIT (options)	Control command that specifies the maximum values for various system resources required by the job. The user should supply the options appropriate to each job. For these options, see the BPM/BP, RT Reference Manual, 90 09 54, or the CP-V/BP Reference Manual, 90 17 64, as appropriate.
3	!LOAD	A control command that loads the modules specified in this record.
	(BI),	Specifies that the first module to be loaded will be on the binary input device.

Example B-3. Merged File from Merge Program Plus User Own-Code, Output on RAD (cont.)

Card	Parameter	Description
	(EF,(MERGE0,:SYS)),	Specifies that the second module to be loaded is an element file (a file under monitor control) named Merge, and is located in the system library. This example assumes that the element file Merge is in the :SYS account. At some user installations, however, it may be in some other account. If it is, substitute that account for the word :SYS in this card.
	(LMN, CHKMRG),	Specifies that the combined load modules will be designated by the name CHKMRG on later control records.
	;	Signals that card 4 is a continuation of card 3.
4	!(PERM)	Specifies that the combined module will be added to the user's library under his account number. (Future uses of CHKMRG will not require the user's binary deck or the LOAD, BIN, or EOD control commands.)
5	!BIN	A control command that specifies the following records will be binary until an EOD command is encountered.
6		The user's compiled or assembled binary deck. Under some conditions, an EOD command is included in this deck.
7	!EOD	A control command specifying that the end of the data (user's program) has been reached and that the following records will be BCD.
8	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGOUT,	Specifies the symbolic name of Merge's output DCB.
	(FILE, NTF),	Specifies that the sorted output is to be placed on the RAD under the name NTF.
	;	Signals that card 9 is a continuation of card 8.
9	!(PASS, MALO),	Assigns the password MALO to the file. This password must be specified by any future user of the file.
	(WRITE, 8)	Specifies that only jobs run under account number 8 may write into the output file.
10	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN1,	Specifies the symbolic name of Merge's first input DCB.
	(DEVICE, 7T),	Specifies that the input file is on a 7-track magnetic tape.
	;	Signals that card 11 is a continuation of card 10.
11	!(INSN, 176, 26, 5373),	Specifies that the input file is contained on reels 176, 26, and 5373.
	(BCD),	Specifies that the input file is written in 7-track BCD mode.
	(IN)	Specifies that this is an input file.
12	!ASSIGN	A control command that conveys file characteristics to Merge's DCBs.
	F:MRGIN2,	Specifies the symbolic name of Merge's second input DCB.
	(DEVICE, 7T),	Specifies that the input file is on a 7-track magnetic tape.
	;	Signals that card 13 is a continuation of card 12.

Example B-3. Merged File from Merge Program Plus User Own-Code, Output on RAD (cont.)

Card	Parameter	Description
13	!(INSN, 123, 45),	Specifies that the input file is contained on reels 123 and 45.
	(BCD),	Specifies that the input file is written in BCD mode.
	(IN)	Specifies that this is an input file.
14	!RUN	A control command that specifies a designated program is to be executed. This command may be used in lieu of a MERGE command.
	(LMN, CHKMRG)	Designates that the module CHKMRG is to be loaded from the user's library and run.
15	!DATA	Signals the monitor that a data deck follows. This command must be used when a RUN command is used to initiate a merge.
16	. FILE (FILES, 2)	Specifies that two input files are to be merged.
17	. REC (783)	Specifies that the input files consist of 783-byte logical records.
18	. BLOCK (3)	Specifies that the input file blocking factor is 3. By default, the output file blocking factor is also 3.
19	. KEYS	Specifies that the input files are to be merged according to certain key fields (three key fields in this case).
	(7, 4, AN, TRAN),	Specifies that key 1 is a four-byte alphanumeric key that starts in byte 7 of each logical record and that the key is to be translated.
	(257, 8, BN),	Specifies that key 2 is an eight-byte binary key that starts in byte 257 of each logical record. By default, the merge is in ascending sequence.
	(90, 1, BN, D)	Specifies that key 3 is a one-byte binary key that starts in byte 90 of each logical record. The merge is in descending sequence.
20	. TRAN (129, 10, A1B1C1D1E1)	Indicates the user-supplied set of character collating sequence values for translation of key 1.
21	!FIN	Signals the end of the job.

Example B-4. On-Line Merge Session (CP-V)

```
!C MRGTST1
1|2|3|4|5|6|7|8|9|10|A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|
```

User requests a copy of the previously sorted file MRGTST1, which consists of 10 single-character records.

Example B-4. On-Line Merge Session (CP-V) (cont.)

```
!C MRGTST2
A
B
C
D
F
G
H
J
K
L
M
N
S
Y
X
Z
```

User requests a copy of the previously sorted file MRGTST2, which consists of 16 single-character records.

```
!SET F:MRGOUT /MRGOUT3

!SET F:MRGIN1 /MRGTST1

!SET F:MRGIN2 /MRGTST2
```

CP-V on-line commands, similar to !ASSIGN in batch, which specify the files MRGTST1 and MRGTST2 as input to, and file MRGOUT3 as output from, the Merge Processor.

```
!MERGE
MERGE SPECIFICATIONS
.REC (1,1)
.REC (1,1)
```

Calls the Merge Processor.
Merge requests input.
Specifies that input and output records are 1-character.

```
.KEYS (1,1)
.KEYS (1,1)
```

Specifies that Merge is to be keyed on one character, in position 1.

```
.FILE (FILES,2)
.FILE (FILES,2)
```

Specifies that 2 input files are to be merged.

```
F\
```

ESCAPE and F keys terminate input.

```
MERGE FILE: 0, RECORDS:          26
MERGE FILE: 1, RECORDS:          10
MERGE FILE: 2, RECORDS:          16
MERGE FILE: 3, RECORDS:           0
MERGE FILE: 4, RECORDS:           0
MERGE FILE: 5, RECORDS:           0
MERGE FILE: 6, RECORDS:           0
MERGE FILE: 7, RECORDS:           0
MERGE FILE: 8, RECORDS:           0
MERGE FILE: T, RECORDS:          26
MERGE SUCCESSFULLY COMPLETED
```

Merge prints statistics of the Merge operation.

```
!C MRGOUT3
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
```

User requests a copy of the Merge output file MRGOUT3.

Example B-4. On-Line Merge Session (CP-V) (cont.)

```
Q
R
S
T
U
V
W
X
Y
Z
1
.
```

CP-V prompts for the next command.

Notes: 1. Underlined characters were sent to the terminal by CP-V; non-underlined characters were typed at the terminal.

2. All terminal input lines were terminated by a carriage return.

XEROX

Reader Comment Form

We would appreciate your comments and suggestions for improving this publication

Publication No.	Rev. Letter	Title	Current Date

How did you use this publication?

☐ Learning ☐ Installing ☐ Sales
☐ Reference ☐ Maintaining ☐ Operating

Is the material presented effectively?

☐ Fully Covered ☐ Well Illustrated ☐ Well organized ☐ Clear

What is your overall rating of this publication?

☐ Very Good ☐ Fair ☐ Very Poor
☐ Good ☐ Poor

What is your occupation?

Your other comments may be entered here. Please be specific and give page, column, and line number references where applicable. To report errors, please use the Xerox Software Improvement or Difficulty Report (1188) instead of this form.

Your name & Return Address

Thank You For Your Interest. (fold & fasten as shown on back, no postage needed if mailed in U.S.A.)

PLEASE FOLD AND TAPE –
NOTE: U. S. Postal Service will not deliver stapled forms

First Class
Permit No. 59153
Los Angeles, CA

BUSINESS REPLY MAIL
No postage stamp necessary if mailed in the United States

Postage will be paid by

Honeywell Information Systems
5250 W. Century Boulevard
Los Angeles, CA 90045

Attn: Programming Publications

Fold

Honeywell

Honeywell Information Systems
In the U.S.A.: 200 Smith Street, MS 486, Waltham, Massachusetts 02154
In Canada: 2025 Sheppard Avenue East, Willowdale, Ontario M2J 1W5
In Mexico: Avenida Nuevo Leon 250, Mexico 11, D.F.

20566, 3C478, Printed in U.S.A.

XS37, Rev. 0