File No. S360-33
Form C24-3321-3

BOS

Systems Reference Library

IBM System/360 Basic Operating System Sort/Merge Program Specifications

This publication describes the IBM System/360 Basic Operating System, Sort/Merge Program. It contains the following information:

1. Minimum machine requirements for sorting or merging records with this program.

2. Program capabilities.

3. A description of the control statements required to define specific sort or merge operations.

4. A description of the facilities provided for inserting user-written routines into the program.

For a list of associated publications and their abstracts, see the IBM System/360 Bibliography, Form A22-6822.

PREFACE

This reference publication describes the IBM System/360 Basic Operating System, Sort/Merge Program. Information is included about the program specifications, including equipment, preparation of control statements, description of the program operation, and procedures for program modifications.

The reader is assumed to have a working knowledge of the System/360. The publications _IBM Basic Operating System/360, Assembler with Input/Output Macros (8K Disk)_, C24-3361, and _IBM Basic Operating System/360, Programmer's Guide (8K Disk)_, C24-3372, are prerequisites.

Fourth Edition, June 1966

This is a major revision of, and obsoletes, C24-3321-2. This revision incorporates the necessary information about the two user exits, 13 and 44, added to the program.

Significant changes or additions to the specifications contained in this publication will be reported in subsequent revisions or Technical Newsletters.

The IBM System/360 Basic Operating System, Sort/Merge Program provides the user with the ability to sort a file of random records, or merge multiple files of sequenced records, into one sequential file. The control-data information can be contained in as many as twelve fields in each record. The records can be sorted or merged into ascending or descending sequence, and an individual sequence can be specified for each control-data field. The output sequence for a merge-only operation must be the same as the input sequence.

The sort/merge program is a generalized program. It must be incorporated into a disk-resident Basic Operating System/360 library because it is designed to run in a disk-resident basic operating system environment. For an introductory discussion of the disk-resident system, see IBM Basic Operating System/360 Programmer's Guide (8K Disk), C24-3372. At execution time, control statements (see Sort/Merge Control Statements) will tailor the generalized sort/merge program to the user's specific application. The sort/merge control statements, along with the required user-prepared job-control statements, are punched into cards and inserted into the card reader with the symbolic address SYSRDR. The sort/merge program, identified by job-control statements, will be retrieved in overlays from the disk-resident library by the control program.

PROGRAM DESCRIPTION

The sort/merge is divided into five phases:

- Assignment Phase (Phase 0)
- Internal Sort Phase (Phase 1)
- External Sort Phase (Phase 2)
- Final Sort Phase (Phase 3)
- Merge-Only Phase (Phase 4)

If sorting is to be done, phases 0, 1, 2, and 3 are executed. If only merging is to be done, phase 0 and phase 4 are executed. Figure 1 is a flowchart of the phases.

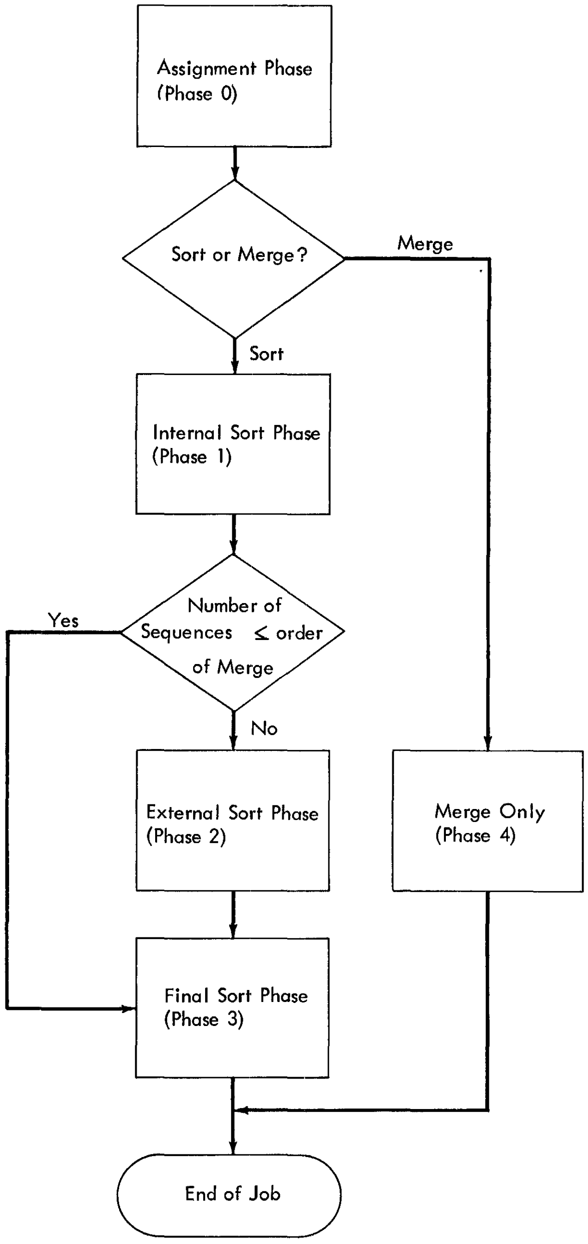

Assignment Phase (Phase 0)
Sort or Merge?
Merge
Sort
Internal Sort Phase (Phase 1)
Number of Sequences ≤ order of Merge
Yes
No
External Sort Phase (Phase 2)
Merge Only (Phase 4)
Final Sort Phase (Phase 3)
End of Job

Figure 1. Flowchart of the Disk Sort/Merge Program Phases

ASSIGNMENT PHASE (PHASE 0)

The assignment phase performs a number of functions. It reads the sort/merge control statements and stores the data from the statements. It performs a diagnostic check on the control information, checking for such things as missing control statements and duplicate or invalid field definers. An expanded version of control statements and parameters are then printed out, if the PRINT option is specified in the OPTION statement (see Sort/Merge Control Statements: OPTION Statement). Next, the assignment phase converts the parameters to binary configurations, and calculates and stores the constants required for the subsequent phases.

If the assignment phase detects certain logical errors in the control statements, the control statements in error must be corrected before the sort run can proceed. If, for example, the user inadvertently specifies a record length that exceeds the specified input block size, the logical error will be detected, and a message will be printed.

INTERNAL SORT PHASE (PHASE 1)

Phase 1 performs the initial sequencing of the input file. The records are read into the main-storage input area and sorted into sequences that are G (the number of records that can be sorted internally at one time) in length. These strings are written out in the 2311 disk-storage work area.

Phase 1 allows multivolume input files. Exits from the program are provided to allow for processing user-prepared subroutines and for checking nonstandard (tape) or user tape or disk labels.

EXTERNAL SORT PHASES (PHASE 2 AND PHASE 3)

The external sort phases merge the ordered sequences produced by the internal sort (phase 1). They perform a two-way to seven-way merge on the strings of records produced by the internal sort phase.

Phase 2 repeatedly merges the strings from the 2311 disk-storage work area until the number of strings is equal to or less than the program-determined order of merge. Phase 3 performs the last pass of the external sort. If the number of strings produced by phase 1 is equal to or less than the order of merge, the first phase (phase 2) of the external sort is bypassed and only the last pass (phase 3) is executed. Phase 3 allows multivolume output files.

MERGE-ONLY PHASE (PHASE 4)

The merge-only phase can be used to merge existing presorted files into one sequential file. A maximum of four tape or disk (not mixed) input files can be merged. The program allows for multivolume input and output. The program also allows for a single file to be reblocked and sequence-checked.

PROGRAM FEATURES

The Sort/Merge Program:

1. Translates mnemonic sort/merge control-card information that describes the file parameters for each input and output file.

2. Sorts a single input file.

3. Allows multivolume input and/or output for tape and disk.

4. Provides for input to a sort operation from punched cards, disk storage [IBM 2311 Disk Storage Drive(s) only], or 7- or 9-track magnetic tapes. Disk input can be distributed over multiple drives.

5. Merges a minimum of one to a maximum of four tape or disk input files.

6. Provides for output to disk storage [IBM 2311 Disk Storage Drive(s) only] or 7- or 9- track magnetic tapes. Disk output can be distributed over multiple drives.

7. Provides for option of writing an output file composed of the disk addresses of the sorted records (ADDROUT option).

8. Provides for specification of an alternate input tape drive (sort operations only) and an alternate output tape drive (sort and merge operations).

9. Provides checkpoint, interrupt, and restart procedures.

10. Provides exits to storage areas for user-written routines.

11. Prints out: The control-card information (optional); record counts at the end of phase 1, phase 3, and phase 4 (optional); and necessary diagnostics.

12. Provides, for sorting, the option to bypass unreadable data blocks when the input file is being read from tape, or to indicate the need for operator intervention when the input file is being read from tape or disk. (The bypass option does not apply to disk input files for the sort operation.)

    A user-exit (EXIT 13) is available to allow the user to print or otherwise analyze the unreadable data block when the input file is being read from tape.

13. Provides for the merge-only operation, the option to bypass unreadable data blocks when the input files are being read from tape or disk, or to indicate the need for operator intervention.

    A user-exit (EXIT 44) is available to allow the user to print or otherwise analyze the unreadable data block when the merge input is being read from tape.

14. Checks the sequence of the records during the final output pass of phase 3 and during phase 4.

15. Permits the user to specify either ascending or descending sequence for each individual control-data field.

16. Processes standard System/360 volume and file labels.

17. Provides exits for user-processing of nonstandard and user labels.

SYSTEM CONFIGURATION

The System/360 used to run the sort/merge program must have at least:

- 8,192 bytes of main storage.
- One IBM 2311 Disk Storage Drive attached to either one multiplexor channel or one selector channel.
- One IBM 1403, 1404, or 1443 Printer; or one IBM 1052 Keyboard Printer.
- One IBM 1442, 2501, 2520, or 2540 Card Reader.

The program supports the following:

- A maximum of 65,536 bytes of main storage.
- A maximum of eight IBM 2311 Disk Storage Drives (including the system drive), two to four of which can be used for intermediate storage.
- One to four IBM 2400-Series Magnetic Tape Units (7 or 9 track) for input/output to a sort operation.
- One to six IBM 2400-Series Magnetic Tape Units (7 or 9 track) for input/output to a merge operation.

The possible combinations of 7-track or 9-track tapes are:

Input	Output
7	7 or 9
9	7 or 9

CONTROL-DATA FIELDS AND COLLATING SEQUENCE

A control-data field is a group of contiguous bytes within a data record, and the data in this field is, in effect, compared with the data in the corresponding field of every record in a file to determine the sorted or merged sequence of the records. The program is capable of sorting or merging records on a maximum of twelve control-data fields with a maximum total length of 256 bytes.

The most significant field, the one with the highest priority, is the major control field, and it will be compared first; the others are minor control fields and will be compared according to their relative priority values. The minor control fields are compared only if the comparisons of the more significant fields result in an equal condition. The individual fields may be contiguous or separated; or, if they contain only unsigned binary data, they may overlap (Figure 2). They may occur anywhere within a data record except in the record length field of the beginning of each variable-length record. A given control field must be located in the same relative position in each record of a file. If, for example, the first two bytes of a fixed-length record in a file are designated as the major control field, the program assumes that the first sixteen bits of data in every record compose the major field for that record.

If a KEY is associated with each disk input record (fixed-length unblocked records only), the KEYLEN entry can be specified in the OPTION statement, and the records can be sorted on the contents of the KEY field (see Sort/Merge Control Statements: OPTION Statement).

Control Data may consist of one of the following:

1. Unsigned binary, including alphameric (character), data.
2. Packed or zoned decimal format.
3. Normalized floating point in either short or long format.
4. Fixed-point format.

The following limitations must be observed when planning the characteristics of the control fields for a file.

- Each control-data field must begin and end on a byte boundary.
- Each control-data field must be at least one byte long.

- Each control-data field may be a maximum of:

  a. 256 bytes for unsigned binary, including character, fields.

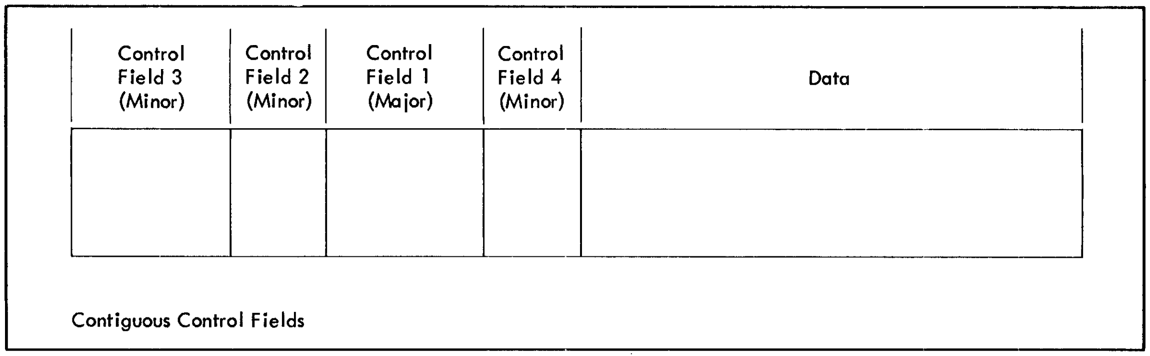
  b. 16 bytes, including a sign, for decimal fields.

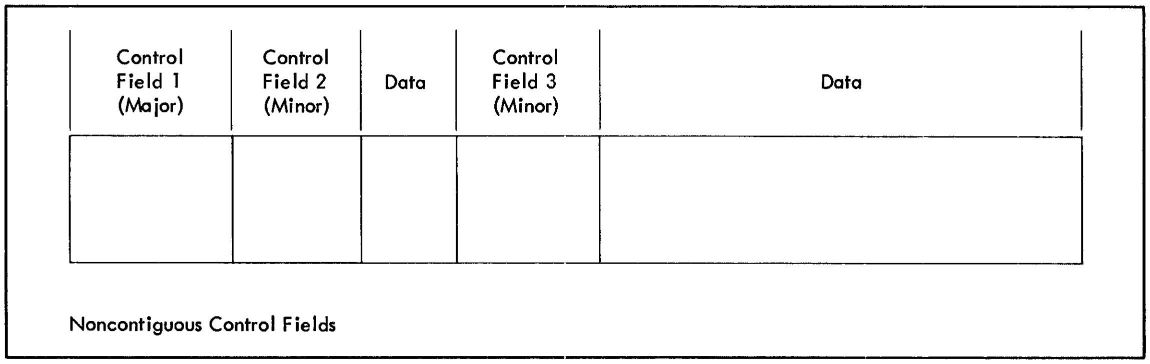
  c. 32 bits (short word) or 64 bits (long word) for normalized floating-point numbers.

  d. 256 bytes for fixed-point fields.

If disk-address output is specified in the OPTION statements, the total length of all control fields plus 8 bytes (CF+8) cannot exceed the input record length.

Files may be sorted or merged into either ascending or descending sequence, and each control-data field can have its individual sequence. For example, a user might specify ascending sequence for the major control field and descending sequence for all the minor control fields. This means that the records will be sorted into ascending sequence on the basis of the contents of control field 1. The remaining control fields will be compared, if necessary, and the records will be sorted into descending sequence within the equals determined by the major control field.





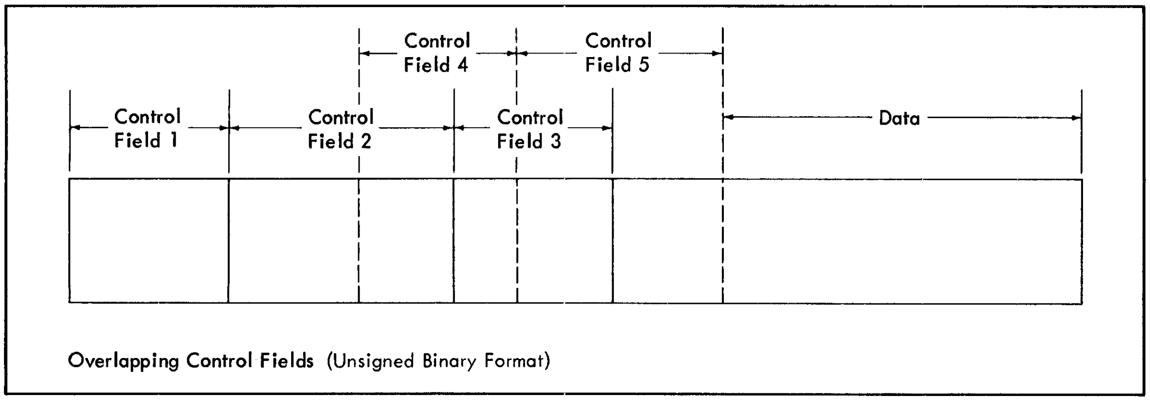


Figure 2. Control-Field Formats

The control-data fields are defined for the program with user-prepared control statements. These statements specify the type of operation (sort or merge), the sequence (ascending or descending), and the size and location of each control-data field (see Sort/Merge Control Statements).

The collating sequence is binary (00000000 to 11111111) for alphameric (character) and binary data. Data conversion routines are included in the sort/merge program to convert, before processing, the following types of data:

Floating Point

Fixed Point

Packed Decimal

Zoned Decimal

to absolute binary so that it can be compared by using the compare-logical instruction. These routines reconvert the data to its original format before the output file is written.

When using either zoned-decimal data or packed-decimal data, the signed fields must be either all ASCII or all EBCDIC; they must not contain a combination of both. If the signed fields are not all ASCII or all EBCDIC, the sort program conversion routines can be bypassed by specifying binary data. If either zoned-decimal data or packed-decimal data are specified, preferred sign codes will be forced. Thus, if the sign code of the input is not the preferred ASCII plus (1010), the preferred ASCII minus (1011), or the preferred EBCDIC minus (1101), a preferred EBCDIC plus (1100) will be forced. The output of the sort will retain these preferred sign codes.

Note: If all records with zoned decimal, packed decimal, fixed point, or floating point data contain the same sign (all plus or all minus) in the same bit configuration, binary can be specified in the SORT or MERGE statement and the sort/merge conversion routines will be bypassed, resulting in a more efficient sort or merge operation. Under these conditions, when records are being sorted or merged on a negative control field, the reverse sequence must be specified in the FIELDS entry of the SORT or MERGE statement. If, for example, a file of records with negative control fields will be sorted into ascending sequence, the following entry would appear in the sort statement:

```
FIELDS=(1,10,D),FORMAT=BI
```

The user has the option of designating a unique collating sequence. By using a programmed exit and a translation table, he can cause the program to sort or merge his records into any collating sequence he wishes to develop. He simply specifies his sequence in a translation table as a set of constants included in the user programming, and enters his translation routine through a programmed exit set aside for this purpose (see EXIT 12 and EXIT 32).

INPUT/OUTPUT SPECIFICATIONS

The sort/merge program can sort or merge fixed-length or variable-length tape or disk records. Tape input files with checkpoint records cannot be processed. It can also sort unblocked punched-card records with a maximum length of 80 characters. The tape or disk fixed-length input records may be blocked in fixed- or variable-length blocks, or they may be unblocked. The tape or disk variable-length input records may be blocked in variable-length blocks, or they may be unblocked (Figure 3).

The most efficient input file configuration is blocked fixed-length records, and the least efficient is variable-length records unblocked. Certain maximums and minimums (control-field length, input/output block sizes, and file size) must be observed by the user when he plans a particular application that includes a sort or merge operation.

The sort/merge program processes disk input files consecutively within the specified extents. When a no-record-found condition is detected on a track, the program will attempt to read from the next consecutive track or from the first track of the next extent, if the condition was detected on the last track of an extent. This does not mean that the input file must have been written consecutively; it means that any file that can be read consecutively (i.e., the records within each track are contiguous and the record numbers are consecutive) is acceptable. Files with indexed sequential organization, for example, cannot be read consecutively by the sort/merge program and, therefore, are not acceptable to the program. Indexed sequential files must be reorganized as consecutive files before being sorted.

RECORD FORMAT

The format of the records, fixed-length and variable-length, to be sorted must adhere to the standards established by the Basic Operating System/360 Input/Output Control System (IOCS). For a complete discussion of record formats which are summarized in Figure 3, see the Systems Reference Library publication IBM Basic System/360, Assembler with Input/Output Macros (8K Disk), Form C24-3361.

Variable-length records can be processed from 7-track tapes with the data conversion feature, 9-track tapes, or 2311 disk storage.

RECORD LENGTHS

The maximum record lengths vary according to the amount of main storage available for sorting.

For sorting, the maximums for fixed-length records are:

Main Storage	8,192	16,384	32,768	65,536
Max. lengths with approx. minimum Supervisor	753	1371	3519	3624
Max. lengths with approx. minimum Supervisor	634	1271	3419	3624

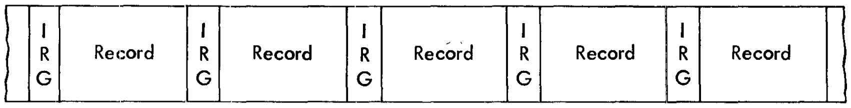
For merge only, the maximums for fixed-length records are:

With approximate minimum supervisor:

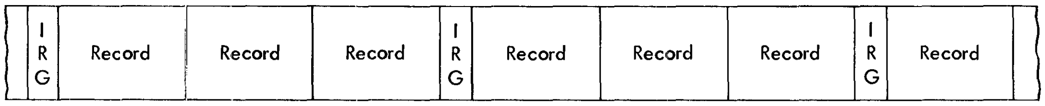
Main Storage	8,192	16,384	32,768	65,536
4-way merge	685	2323	3624	3624
3-way merge	856	2904	3624	3624
2-way merge	1,142	3624	3624	3624

Tape

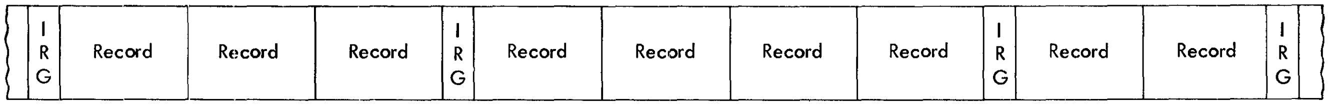
Unblocked Fixed-Length Records (Input/Output)



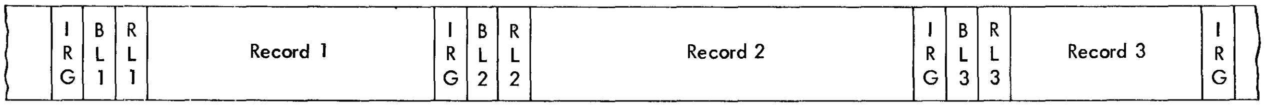
Fixed-Length Records in Fixed-Length Blocks (Input/Output)



Fixed-Length Records in Variable-Length Blocks (Input)



Unblocked Variable-Length Records (Input)



BL (Block Length) = The number of bytes between interrecord gaps, which include the four bytes required for the block-length field plus the four bytes required for each of the record-length fields. (4 + 4 + length of record = contents of BL).

RL (Record Length) = The number of bytes in the record length added to the four bytes required for the record-length field (4 + length of record = contents of RL).

Blocked Variable-Length Records (Input/Output)

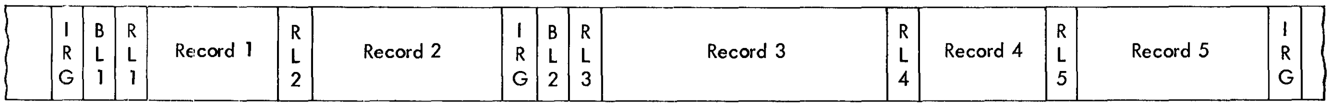


Figure 3. Input/Output Blocking Formats (Part 1 of 2)

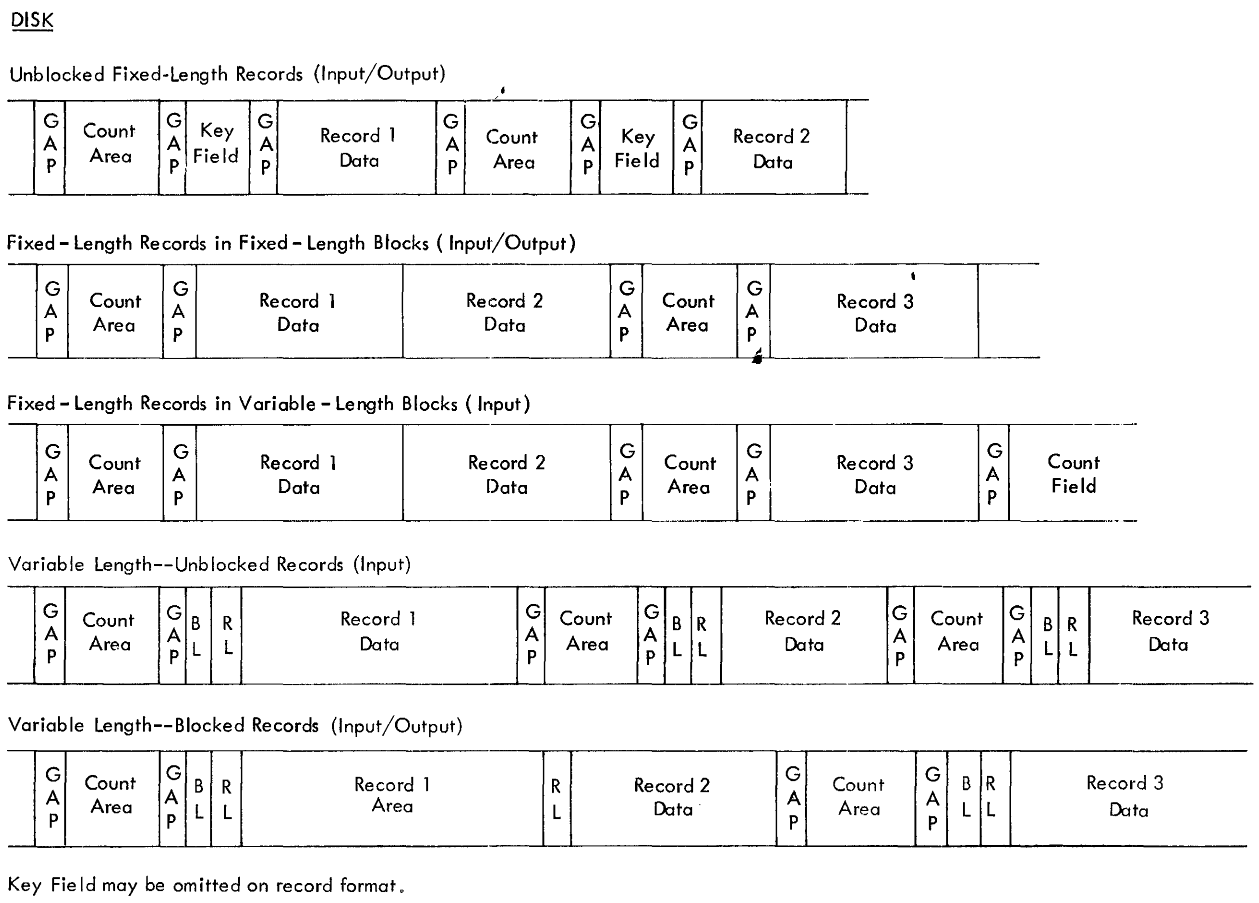

DISK
Unblocked Fixed-Length Records (Input/Output)
GAP
Count Area
Key Field
Record 1 Data
Record 2 Data
Fixed-Length Records in Fixed-Length Blocks (Input/Output)
Record 3 Data
Fixed-Length Records in Variable-Length Blocks (Input)
Count Field
Variable Length--Unblocked Records (Input)
BL
RL
Variable Length--Blocked Records (Input/Output)
Record 1 Area
Key Field may be omitted on record format.

Figure 3. Input/Output Blocking Formats (Part 2 of 2)

With approximate maximum supervisor:

Main Storage	8,192	16,384	32,768	65,536
4-way merge	566	2163	3624	3624
3-way merge	707	2704	3624	3624
2-way merge	943	3606	3624	3624

These figures for sorting and merging with a minimum and maximum supervisor are based upon the following assumptions:

1. Unblocked fixed-length records.

2. No data translation.

3. No user programming.

4. One control field.

5. Approximate minimum physical IOCS and supervisor size of 3,500 bytes.

6. Approximate maximum physical IOCS and supervisor size of 4,096 bytes for 8,192 bytes of main storage and an approximate maximum size of 4,300 bytes for 16,384 and above of main storage. (The supervisor and IOCS assembled to handle the sort program must not exceed 4,096 bytes, including the patch area in an 8K system.)

For operations involving blocked records, variable-length records, disk-address output, or for operations that do not adhere to the assumptions upon which the preceding maximums are based, calculate the maximum input/output block lengths with the formulas in Appendix C.

The minimum number of records that can be contained in a single input or output block is 1 record. Twelve bytes is the minimum length for a tape input/output record or block. One byte is the minimum length for card or disk record or block.

LABEL PROCESSING

The sort/merge program processes standard IBM System/360 header and trailer labels as part of its input/output processing. Nonstandard label processing (reading, writing, and checking) is permitted with tape input/output only, and is the responsibility of the user (see User-Prepared Routines). For a complete discussion of standard label processing, see IBM Basic Operating System/360 Programmer's Guide (8K Disk), Form C24-3372.

STANDARD LABELS

A volume label must be present if standard labels are to be checked. A volume label is used to identify a physical storage unit such as a disk pack or reel of magnetic tape.

One volume label is the minimum that must appear on a volume of records, but the user may have as many as seven additional volume labels. The sort/merge program will check the first three fields of the first volume label only. All additional volume labels will be bypassed. Existing volume labels are never overwritten.

If standard label processing is specified, the program will check the Volume Serial Number (field 3) on input tape or disk files.

A header label identifies the logical file (within a volume) to be processed, and the sort/merge program will:

- Check the first input header on each tape or disk input volume, and write the first output header after the existing volume labels on each output tape volume or in the VTOC on each output disk volume.
- Read and allow the user the option of checking additional user input headers for disk and tape; write after allowing the user to create additional user output headers for tape and disk.
- Check for a tape mark (with tape input) after the last header on each input file, and write a tape mark (with tape output) after the last header on each output file.
- Check the Expiration Date (field 10) on tape or disk output files.

A trailer label indicates the logical end of file or the end of a volume of that file. The sort/merge program will:

- Check the first tape input trailer on each input volume for EOV or EOF. It will read and allow the user the option of checking additional tape user input trailers. It will write the first output tape trailer, and write, after allowing the user to create user output tape trailers.
- Write a tape mark (with tape output) before the first trailer on each output file.

The user may include as many as eight additional header (tape or disk) and eight additional trailer labels (tape only). All additional header and trailer labels must be 80 characters long. For disk and tape labels, the first four bytes must contain standard information.

IOCS, for user disk header labels, creates the first four identification characters. The letters UHL are placed in the first three bytes of the label. The label number is placed in the fourth byte. User header labels range from UHL1 to UHL8.

For tape, the user must create his own header and trailer label identification in the first four bytes of each label. The letters UHL must be placed in the first three bytes of the header label. The label number must occupy the fourth byte. The letters UTL must be placed in the first three bytes of the trailer label and the label number must occupy the fourth byte. User header and trailer labels for tape also range from UHL1 to UHL8 and from UTL1 to UTL8.

The remaining 76 bytes may contain whatever information the user wants. IBM standard headers and trailers are identical in format, except for the first field.

The information required for checking the initial input header or creating the initial output header is provided to the sort/merge program in job-control statements. The program will verify the expiration date and then write a standard-format header label on the output file.

Although an output tape must always have a volume label, it may or may not contain a standard header and trailer label. If a tape is used without a header label, the program will create and write the first header label after the existing volume label. If an output header exists, the program will check the expiration date and either indicate nonexpiration or write a new header label.

NONSTANDARD LABELS AND UNLABELED FILES (TAPE ONLY)

Programmed exits are provided for the user to read and check nonstandard tape-input headers and trailers, and to compile and write nonstandard tape-output headers and trailers. If the user specifies nonstandard input labels, but does not specify the accompanying exits, the program will search for the tape mark that must separate the labels and the data. By searching for the tape mark, the program will bypass the nonstandard labels. If the user specifies nonstandard input labels, and specifies the accompanying exits for processing these labels, a tape mark may or may not immediately precede the first data record. It is the user's responsibility to position the tape after the last nonstandard label.

Unlabeled tape files can be processed by the program. The first record read from an unlabeled input volume can be either a tape mark or a data record. If it is not a tape mark,. the program will backspace the tape to the beginning of the first record.

A tape mark can be written before the first data record and one will be written after the last data record on each output volume with nonstandard labels or no labels.

INPUT/OUTPUT DEVICES

Certain considerations must be made in the use and arrangement of input/output devices, and certain assumptions are made by the program.

Assignment of I/O devices to specific channels, if a 2-channel system is used, is left to the discretion of the user. The sort/merge program does not require a particular device on a particular channel.

The program-required disk work area can be composed of a maximum of four individual extents. The total number of tracks required for a work area (the sum of the extents) must not be greater than the total number of usable tracks on four disk packs.

DISK DRIVES

Input to a sort operation can be read from multiple disk drives, and output can be written on multiple disk drives. The input and output areas are flexible (multiple extents can be specified) and can be arranged around the program-required work area, which must be on-line at all times during execution; but the eight-drive program maximum must be observed.

Input to a merge operation can be read from multiple disk drives, and output can be written on multiple disk drives, but the eight-drive program maximum must be observed.

The user is free to assign any symbolic name he wishes to the disk drives used for sorting or merging, providing he observes the assignment limitations established by the disk-resident control program (see IBM Basic Operating System/360 Programmer's Guide (8K Disk), Form C24-3372.

TAPE DRIVES

Primary and alternate input/output tape drives can be specified to eliminate the time required to rewind and unload the tape reel and mount the new reel for the input or output process when sorting a tape file with multivolume input or output. If the input records are contained on more than one volume, the next volume can be made ready on the alternate input drive while input records are being read from the current drive.

The program assumes, if tape input is specified, that the initial reel of input to be sorted will be read from the tape unit with the symbolic address SYS002, and the alternate input unit will be the one with the symbolic address SYS003. If a multivolume input file will be read from a single tape drive, the symbolic addresses SYS002 and SYS003 must both be assigned to the same tape drive. If tape output is specified, the program assumes that the output will be written on the unit with the symbolic address SYS000. The alternate output drive is assumed to be SYS001. If a multivolume output file will be written on a single tape drive, the symbolic addresses SYS000 and SYS001 must both be assigned to the same tape drive.

If tape input is specified for a merge-only operation, the units with the symbolic addresses SYS002 through SYS005 are assumed to contain input files 1 through 4, respectively. If, for example, three tape files will be merged, the program assumes they reside on SYS002, SYS003, and SYS004. If tape output is specified, the program assumes the initial output drive is SYS000 and the alternate output drive is SYS001.

CHECKPOINT, INTERRUPT, AND RESTART

The checkpoint, interrupt, and restart feature permits the user to interrupt the sorting process during phase 2 or phase 3 and conveniently restart it. This feature can be used during sort runs only; it will not function in a merge-only operation. The sort program automatically writes a checkpoint record on disk during each pass of phase 2. At this point in the program, the sorted records are being merged into strings that increase in size with each merge pass. The checkpoint record is composed of the information needed to restore the program at the beginning of the interrupted pass. Processing can then continue from this point.

To interrupt the program, the user presses the system-reset key. If the program is interrupted during phase 1, the entire program must be rerun, and all the processing that has taken place must be repeated. If the program is interrupted at the beginning or during phase 2, or phase 3, the user must punch an OPTION statement that contains the RESTART field definer (see Control Statements: OPTION Statement). Printed messages will indicate the beginning of phase 2 and the beginning of phase 3. All the control statements that were used to define the initial sort operation must be reread by the card reader.

When the assignment phase processes the RESTART entry in the control statement, it transfers control to phase 2, which extracts the information from the last disk checkpoint record. The processing that occurred prior to the interrupted merge pass is bypassed. Processing is restarted at the beginning of the interrupted merge pass.

Note: It is imperative that each disk pack be returned to a 2311 disk drive with the symbolic name it had when the program was interrupted, and the disk storage areas being used by the sort at the time of interruption must contain the data that was present when the interruption occurred. The work area extent cards (SORTW cards) must have, at restart time, the same sequence number they had when originally read.

If the program is interrupted in phase 3 after the sort program has written a standard output label and has begun to write the final output, a halt will occur in the restart run when an unexpired date is encountered in this output label.

If the output area is the same one specified previously, the user should ignore the message for this halt and continue processing.

CHECKING FEATURES

Two sort/merge checking features are included in the program: a record-count feature, and a sequence-check feature.

The record-count printout must be specified by the user; it is not automatic. Including the field definer PRINT in the OPTION statement (see Control Statements: OPTION Statement) will cause a count of the processed records to be printed out at the end of phase 1, phase 3, and phase 4.

A sequence check, the other checking feature, occurs automatically during phase 3 and phase 4. The function of the sequence check is to make sure the output records are in the specified sequence. If a sequence error is detected, the system prints a message and then enters the wait state.

USER-PREPARED ROUTINES

Provision is made in phase 1, phase 3, and phase 4 of the program for the addition of subroutines written by the user.

These subroutines must be cataloged in the core image library in the systems residence. All the subroutines for a particular phase are considered to be a single program phase and are handled as such by the supervisor. This means that each set of routines must be labeled with a unique label. The MODS statement (see Sort/Merge Control Statements) is used to specify the name of a set of subroutines for a given phase, and this set will be retrieved using the name.

It is recommended that this name be five characters in length, the first four being DSOR and the last being any character the user assigns to identify this as a user phase name. Job Control will search the core image library for all routines identified as DSOR and will construct a phase dictionary of them. Therefore, by having such a dictionary, the FETCH time is reduced, making the sort/merge program run more efficiently.

The interface for the sort/merge program and the user subroutines is designed to allow user modifications to be compatible with future versions of the program. The program links to these subroutines through use of branch instructions (BAL), called exits, that are part of the program.

The program contains the following exits:

- Phase 1 -- EXIT 11, EXIT 12, EXIT 13

- Phase 3 -- EXIT 31, EXIT 32
- Phase 4 -- EXIT 41, EXIT 42, EXIT 43, EXIT 44

If these exits are not used, the program passes over the branch instruction and goes on to execute the next sequential instruction. To activate the exits, the user must make the required entries in the MODS control statement (see Sort/Merge Control Statements). He must also assemble a branch table at the origin of the subroutines for each phase. The origin address specified in the PHASE card for the combined subroutines in a single phase should be the address that results when the length of the subroutines plus the branch table is subtracted from the main storage capacity specified in the STORAGE entry of the OPTION statement. The origin address must be adjusted to the next lower double-word boundary. The starting location for all programs to be assembled must be a double-word boundary.

If, for example, a sort operation will be run on a system with 8,192 positions of main storage, and the phase-1 user programming (branch table plus routines) requires 300 positions of main storage, the resultant address, when the length of the user area is subtracted from the system capacity, is 7,892. Adjusted to a double-word boundary, the origin address is 7,888.

If the user does not adjust the origin address of 7,892, the linkage editor, at catalog time, automatically will adjust the origin address at the next higher double-word boundary of 7,896. When the user's routine is fetched into main storage, the new origin address would cause the loading of the user's routine to continue beyond the limits of main storage. This error would be indicated by the supervisor.

The program, after encountering the MODS entry during the assignment phase, will branch to the branch table when it encounters the exit; and the branch table will cause another branch to the first instruction of the associated subroutine. The last instruction of the subroutine must be a branch back to the sequential instruction that follows the exit in the main program.

The two general registers involved in the exits are registers 14 and 15.

The branches function in the following manner. The exit instruction is BAL 14,X(15). This instruction stores the address of the next sequential instruction in register 14, and the branch address is formed from the constant base address stored in register 15 plus X (X=either zero or a multiple of 4). Register 15, which is initialized by the sort/merge program, must be specified as the base register in the USING statement in all user-assembled subroutines. Insert the statement USING *,15 before the first branch-table entry when assembling the user subroutines for each phase.

If the user chooses to activate one or more exits in any phase, he must prepare, immediately ahead of his routine, a branch table that includes a 4-byte entry for each branch up to and including the last exit to be used in that phase.

The first entry in the user-prepared branch table must apply to the lowest number exit in that phase of the sort/merge program. The second entry must apply to the next lowest numbered exit in that phase. The third entry must apply to the next lowest numbered exit in that phase. The fourth entry, which is possible only in phase 4, must apply to the next lowest numbered exit (EXIT 44) in that phase of the sort/merge program. B 0 (14) instructions should be used to fill the unrequired en-entries. If, for example, he chooses to use exit 32, he should include a B 0 (14) instruction as the entry for exit 31 if it is not used. The entries for the unused exits that follow can be ignored. The label of the first instruction of his subroutine might be LABEL2, although he is free to assign any label he chooses. For the example just cited, the branch table would contain:

B 0(14)=	Branch table entry for exit 31.
B LABEL2 =	Branch table entry for exit 32.

The first instruction of this subroutine would be labeled LABEL2, and a final instruction would branch to the address in register 14.

```
LABEL2.
     .
     .
     .
     .BR 14
```

Part of the sort initialization for a phase will be the FETCH of the user routine for that phase. The FETCH will load the user routines into main storage and pass control to the user routine at the point designated by the user at assembly time of the phase. At this location the user can give a BR 14 instruction to pass control to the sort/merge program or perform some user program initialization of his routine and then give a BR 14 instruction to return control to the sort/merge program.

Additional considerations must be made by the user who will use the programmed exits to add his subroutines. It is to the user's advantage to limit his programming to the smallest possible main-storage area, because the size of the control program and sort-program phase and the size of the user's subroutines determine the amount of main storage available for processing data records. During the assignment phase, the number of storage bytes required by the control program and the sort program is added to the number of bytes required by the user's subroutines, and the total is subtracted from the size of the system. The remaining storage is then allocated to input/output areas, and the speed of the program can be increased by the increased size of the input/output areas.

Also, the user's subroutine must provide the programming to store the contents of any register (registers 0 through 11 and register 14) that will be used by the subroutine and then restore these contents before the branch back to the sort program is executed, except in the case where specific instructions are given within the exit write-up (registers 8 and 9).

EXIT 11 (E11)

This exit is available during phase 1. It can be used to check user (UHL) input headers (tape or disk), and trailers (tape) or to read and check nonstandard input headers and user and nonstandard trailers (tape only). IOCS checks standard input-tape trailers for EOV or EOF designations only. For a complete discussion of nonstandard label and user label processing, see IBM Basic Operating System/ 360 Programmer's Guide (8K Disk), Form C24-3372.

User Labels

User labels are read into the IOCS label area by IOCS. Control is then passed through the sort/merge program to the user for execution of his checking subroutine. When this exit occurs, the label is in the IOCS label area, and the low-order byte in register zero contains an O for an OPEN condition, a V for an EOV condition, or an F for an EOF condition. Register 1 contains the address of the IOCS label area. Re-entry to the sort/merge program must be made with the LBRET macro. For further information on re-entry see the discussion of the LBRET macro in IBM Basic Operating System/360, Assembler with Input/Output Macros (8K Disk), Form C24-3361.

If standard labels are specified and are followed by user labels but this exit is not specified, the standard labels are processed and the user labels are bypassed. If the input file is read from tape, the program will search for the tape mark that precedes the first data record.

Nonstandard Labels

If nonstandard input-tape labels will be processed through this exit, the user is responsible for designating an area of main storage that can be used to read and check them. He must also provide the programming to perform these functions. The EXCP macro must be used in reading these labels. When a file is opened, all the nonstandard headers must be processed or bypassed; otherwise the program will assume the labels are data records. The sort program will process the tape mark following the header labels.

When this exit occurs and nonstandard labels have been specified, the low-order byte of register zero contains an O for an OPEN condition, a V for an EOV condition, or an F for an EOF condition. The two low-order bytes of Register 1 will contain the symbolic unit number of the device from which the label will be read. This unit number (a two-byte hexadecimal representation of the symbolic unit for the I/O device) must be inserted into the symbolic unit address bytes of the user CCB (see IBM Basic Operating System/360, Assembler with Input/Output Macros (8K Disk), Form C24-3361. Re-entry to the sort/merge program must be made through register 14 (BR 14).

If nonstandard labels are specified but this exit is not specified, the program will read from the input tape until it encounters a tape mark. This tape mark must precede the first data record.

EXIT 12 (E12)

This exit is available in phase 1 just before each logical record in the input area is about to be processed. The address of the high-order position of the record to be processed will be available in register 5.

It can be used for translation of data not already in the form specified in the control card. If data records will be translated, the user must provide, as part of his programming, the translation table. This exit can, in addition, be used for modification of the data within each record. Records cannot be lengthened, shortened, or deleted.

Data-conversion routines are included in the sort/merge program, but are not accessible to the user for his programming. These routines convert the following types of data after exit 12 occurs, but before the records are sorted.

Fixed Point -- The sign bit is complemented on input and returned to its initial state on output.

Packed Decimal -- If the number is positive, the sign is moved to the high-order half-byte of the field, and the field is shifted to the right by one-half byte. If the number is negative, the same procedure is followed but the entire field is complemented.

Zoned Decimal -- In the byte containing the sign, the sign and the digit are reversed, and the procedure for packed decimal is used.

Floating Point -- If positive, the sign will be changed. If negative, the entire field is complemented.

The data is reconverted to the data format specified in the control card before exit 32 occurs.

EXIT 13 (E13)

Exit 13 is available during phase 1 after the program has determined that an input tape block is unreadable. This exit can be used for output of the data block, using the EXCP macro instruction, which allows output on all devices supported by the Supervisor. The address of the high-order position of the block is contained in register 7.

When control is returned to phase 1, the unreadable data block is bypassed, because the unreadable characters, after being forced to odd parity by the hardware, may not be the same as the original input.

EXIT 31 (E31)

This exit is available during phase 3. It can be used to create and write nonstandard output tape headers and tape trailers, or to create user tape or disk headers and tape trailers.

User Labels

User labels can be created in the IOCS label area and will then be written by IOCS. When the exit occurs, the IOCS label area is available to the user, and its address is in register 1. The low-order byte in register zero contains an O for an OPEN condition, a V for an EOV condition, or an F for an EOF condition. IOCS will write the label out and pass control to the user after each label is written. When each user label has been created, re-entry to the sort/merge program must be made with the LBRET macro. For further information on re-entry see the discussion of the LBRET macro in IBM Basic Operating System/360, Assembler with Input/Output Macros (8K Disk), Form C24-3361.

Nonstandard Labels

If nonstandard labels will be processed through this exit, the user is responsible for designating an area of main storage that can be used to create labels. He must create the labels and write them. The EXCP macro must be used in writing the labels.

When the exit occurs, the low-order byte of register zero contains an O for an OPEN condition, a V for an EOV condition, or an F for an EOF condition. Re-entry to the sort/merge program must be made through register 14 (BR 14) after all nonstandard labels have been created and written. Register 1 will contain the symbolic unit number of the device on which the label is to be created.

EXIT 32 (E32)

This exit is available during phase 3 after each record has been moved to the output area before being written into the output file. It can be used for insertion, deletion, alteration (including record shortening), substitution, summarization, or for data translation. The sorted record (L1 - see Sort/Merge Control Statement: Record Statement) is moved in its entirety (except when the ADDROUT option is chosen) to the output area. When EXIT 32 occurs, register 8 contains the address of the high-order byte of the current output record (record just moved to the output area). The current output record is equal to L3 or its equivalent.

Two methods are available by which a record can be inserted by the user. (1) He may insert his record immediately following the current output record, or (2) he may insert his record in the area occupied by the current output record, and then insert the current output record behind his inserted record. A variation to the second method is also available. The user may substitute his record in the area occupied by the current output record. In either case, multiple insertions, handled one at a time, are allowed.

When using the first method, the user must save the contents of register 9 in his user area. He then loads into register 9, the high-order address of the record he wants to insert. The sort program is then reentered with a B 4(14) instruction. After inserting the record, the sort program will return control to the user's program. If another record is to be inserted (multiple insertions), the user simply reloads register 9 with the high-order address of the next record and re-enters the sort program by using a B 4(14) instruction. This process is repeated until no more records are to be inserted. At this point, the user must restore register 9 to its original contents and return to the sort program by a BR 14 instruction.

When using the second method, the user is responsible for storing and saving the current output record in his own programming area. He also must store and save the contents of register 9. The high-order address of the record to be inserted in place of the current output record can be moved into register 9. The address of the current output record is available in register 8. Before re-entering the sort program, the user must update register 8 by subtracting from it the length of the record that was saved, not the length of the record to be inserted. Control is returned to the sort program with a B 4(14) instruction. The program will insert the user's record, and it will return control to the user's program. If more records are to be inserted, follow the procedure for multiple insertions. When the insertions to be made at this point have been completed, the user must then handle the output record, which he saved, like an insertion. After this record has been re-inserted, control will return to the user. At this point, the value contained in register 9 when the exit occurred, must be restored by the user, and the sort program must be re-entered with a BR 14 instruction.

In using the variation to the second method, the user saves the contents of register 9. The high-order address of the record to be inserted in place of the current output record can be moved to register 9. The user must decrease the contents of register 8 by the length of the record being deleted. The sort program is re-entered by a B 4(14) instruction. Multiple insertions are handled as in the second method. Register 9 must be restored by the user and the sort program must be re-entered with a BR 14 instruction after all insertions have been completed. Except in the case of insertions, all re-entries to the sort program are made with a BR 14 instruction. If variable-length records are inserted, the first four bytes of the inserted record must be the record-length field. The length must be a value less than or equal to the maximum output record length (L3 - see Sort/Merge Control Statements: RECORD Statement).

If fixed-length records will be inserted, and truncation has been specified (the specified L3 value is less than the specified L1 value), the length of the inserted record must be equal to the output record length (L3).

If the current output record is to be deleted by the user, the contents of general register 8 must be decreased by the length of the deleted record (L3 for fixed-length records, or the contents of the record-length field for variable-length records). The sort program must be re-entered with a BR 14 instruction.

Record alteration can be performed in the sort output area, and the address (high-order byte) of the record to be altered is contained in register 8. If the user has specified record shortening for fixed-length records by specifying varying lengths in the LENGTH definer of the RECORD statement, he need not use this exit. The program will truncate the record from right to left according to the user's specification of L3. If the record is to be shortened in any way other than by removing the low-order bytes, the user should perform the alteration in his own programming area. For variable-length records, the user must update the record-length field to reflect the new record length, and he must subtract from register 8 the number of bytes deleted from the record. If the altered record has been moved to a user alteration area, the user has two methods of restoring the record into the output area. He can move the record himself and return to the program by a BR 14 instruction; or he can follow the insertion procedure and return by a B 4(14) instruction.

The last record has been moved to the output area when general register 10 contains a negative value. The user can test for this condition with two instructions:

```
LTR 10,10
BM  Last Record Processed Routine
```

Before EXIT 32 occurs, the sort program will have sequence checked the record moved to the output area, and the control data will have been converted to the format specified.

The sort/merge program keeps a count of the records processed during phase 3. This count, however, does not take into consideration those records inserted or deleted by the user at EXIT 32.

EXIT 41 (E41)

This exit is available during the merge phase of a merge-only operation. It can be used to check nonstandard or additional tape-input headers, and to create and write nonstandard or additional tape-output headers and trailers. It can also be used to check disk-input user headers, or to create disk-output user headers. The sort/merge program checks standard input trailer labels (on tape) only for EOV or EOF designations. See EXIT 11 and EXIT 31 for additional information on label processing.

When the exit occurs, the low-order byte of register zero contains an O for an OPEN condition, a V for an EOV condition, or an F for an EOF condition. Register 1 will contain the address of the IOCS area for user labels, and register 2 will contain the symbolic unit number of the device on which the label was encountered. The sort/merge program opens the merge-output file (FILEO) first, and then opens the input files, FILEA, FILEB, FILEC, and FILED, respectively.

EXIT 42 (E42)

This exit is available in the merge phase of a merge-only operation just before each record in the input area(s) is about to be processed. The address of the high-order position of the record to be processed will be available in register 9.

This exit can be used for translation of data not already in the form specified in the control card. If data records will be translated, the user must provide, as part of his programming, the translation table. This exit can, in addition, be used for modification of the data within each record. Records cannot be lengthened, shortened, or deleted.

EXIT 43 (E43)

This exit is available during the merge phase of a merge-only operation. It functions exactly like EXIT 32 and can be used for the same purpose. For programming details, see EXIT 32.

EXIT 44 (E44)

Exit 44 is available during the merge phase of a merge-only operation after the program has determined that an input tape block is unreadable. This exit can be used for output of the data block, using the EXCP macro instruction, which allows output on all devices supported by the Supervisor. The address of the high-order position of the block is contained in register 1.

When control is returned to phase 4, the unreadable data block is bypassed, because the unreadable characters, after being forced to odd parity by the hardware, may not be the same as the original input.

Standard job-control cards are required to define a sort or merge operation to the Job Control Program. For a complete discussion of the job control cards and their formats, see <u>IBM Basic Operating System/360 Programmer's Guide (8K Disk)</u>, Form C24-3372. The following discussion contains only the type of cards required to sort or merge and the characteristics of these cards that make them unique to a sort or merge operation. It does not attempt to explain card formats or sequences.

SORT STATEMENTS

For a sort operation, include the following cards, if they are applicable.

<u>General</u>

JOB--DSORT is the required operand entry.

ASSGN--Assign cards are required only if the I/O devices have not been previously assigned the appropriate symbolic names required by the sort program.

DATE--A date card is required only if it was not previously made available to the system.

<u>Input File (required only for standard labels)</u>:

VOL--A volume card is required to define the input file. The file name punched into this card must be SORTI. Symbolic unit names are discussed in the section <u>Input/Output Devices</u>. The symbolic unit name assigned to the first device from which input will be read must be punched in this card. Only one volume card is required for the input file.

DLAB or TPLAB--A disk label card (for disk input) or a tape label card (for tape input) must be included.

XTENT--For disk input only, one extent card must be used to define the high and low limits of each disk area from which input records will be read. If, for example, the input file is broken and stored in two different areas, two extent cards, one for each area, must be included. The symbolic unit name of the device to which the extent applies must be punched in the card.

<u>Work Area</u>:

VOL--A volume card is required to define the sort work area on disk. The file name punched into this card must be SORTW. The symbolic unit name assigned to the first drive on which a work file will be written must be punched in this card. Only one volume card is required for the work area, although the work area may be composed of as many as four diverse disk areas.

DLAB--A disk label card is required. Only one label card is required for the total work area.

XTENT--One extent card must be used to define the high and low limits of each disk area that will be used as part of the work area. The total work area required need not be a single disk area. It can be as many as four disk areas which, when taken as a whole, are equivalent to the required total. One extent card is required, but as many as four can be included. The last extent card specified for the work area must contain a minimum of four tracks. The symbolic unit name of the device to which the extent applies must be punched in the card. The extent cards must be placed in the card reader so that the symbolic names are in an ascending sequence. For example, if the user wishes to specify two work areas, one on SYS001 and the other on SYS002, the extent cards for the work area on SYS001 must be first and SYS002 second. The extent-sequence number, however, governs the order in which these work areas are processed. Therefore, if the user wishes to process the work area on SYS002 first, he would punch 001 in the sequence-number field of the extent card for SYS002 and 002 in the sequence-number field of the extent card for SYS001.

<u>Output File (required only for standard labels)</u>:

VOL--A volume card is required to define the output file. The file name punched into this card must be SORTO. The symbolic unit name assigned to the first device on which output will be written must be punched in this card. Only one volume card is required for the output file. The assumed symbolic name, with tape output, is discussed in the section Input and Output Tape Drives.

DLAB or TPLAB--A disk label card (for disk output) or a tape label card (for tape output) must be included.

XTENT--For disk output only, one extent card must be used to define the high and low limits of each disk area into which output records will be written. The symbolic unit name of the device to which the extent applies must be punched in the card.

General:

EXEC--This card is required.

The job control statements are followed by the sort/merge program control statements (see Sort/Merge Control Statements).

MERGE STATEMENTS

The job-control cards required for a merge-only operation are governed by the same conditions as the cards required for a sort operation. The required operand entry for the merge-only job control cards is DSORT. Each file must be defined in the same relative way with the same type of control card. Only the following exceptions exist:

1. The files must be named with these alphabetic designations.

File	Name
First Input	FILEA
Second Input	FILEB
Third Input	FILEC
Fourth Input	FILED
Output	FILEO

2. Tape files are assumed to reside on the tape drives with the following symbolic names (see Input and Output Tape Drives).

File	Symbolic Name
First Input	SYS002
Second Input	SYS003
Third Input	SYS004
Fourth Input	SYS005
Output	SYS000
Alternate Output	SYS001

Note: For merge-only operations, the program requires two consecutive disk tracks to store the label-processing routines and other information that must be stored on disk temporarily. The user must define this area as a minimum of a two-track work area. The file name punched in the work area volume card is MERGEW. (No SORTW card is required for a merge-only operation.)

SORT/MERGE CONTROL STATEMENTS

Sort/merge control statements are necessary to define the user's specific sort or merge operation. It is the user's responsibility to provide accurate control information that describes the files to be sorted or merged, the control-data fields upon which the records will be sorted or merged, the program options to be used, and the modifications to be made to the program. This information is punched into control-statement cards, and the cards are inserted into the card-read device assigned the symbolic name SYSRDR. During the assignment phase, each control statement is analyzed and validity checked for incorrect entries and inconsistent combinations of entries. If any errors are detected, the program prints a message that indicates the nature of each error upon completion of the control-statement analysis.

Each control-statement card can contain a maximum of one control statement, although a single statement may require more than one card. The cards must be punched in the following format:

Column: 1

Contents: Leave blank.

Columns: 2 - 71

Contents: Parameter List.

Column: 72

Contents: Continuation Column. If the control statement extends from one card to another, this column must be punched with any character to indicate that the following card contains a part of the statement.

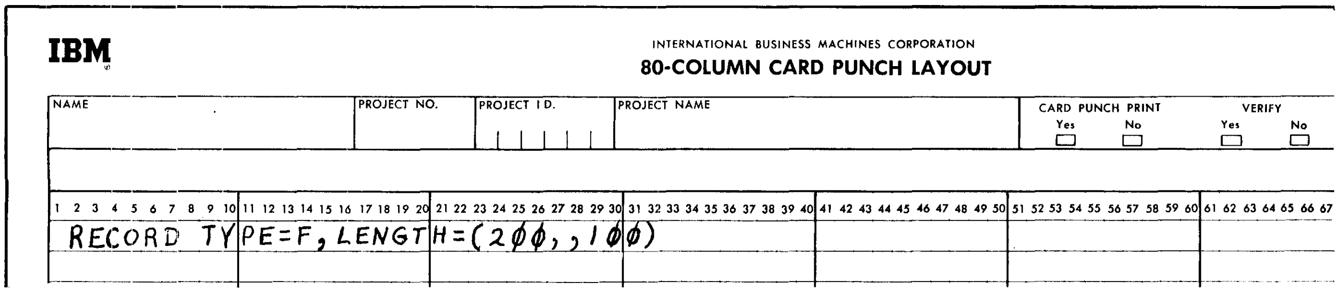
A continuation card (i.e., a card that follows a card with a character punched in column 72) must be blank from column 1 through column 15. The parameter list must continue in column 16.

Columns: 73 - 80

Contents: User Identification. These columns may be used to identify the sort/merge application defined by the control-statement cards.

The parameter list (columns 2-71, or 16-71 in a continuation card) can contain three kinds of entries: a statement definer (required for each statement, a field definer (required for each statement), and values (sometimes required to complement the field definer). A statement definer is a mnemonic entry (SORT, MERGE, RECORD) that identifies the nature of the information in the statement. For example, the RECORD entry denotes a statement containing information about the data records. The statement definer is always the first entry in a control-statement card (unless the card is a continuation of the statement from the preceding card, in which case the preceding card must have a character punched in column 72). A field definer is a mnemonic entry that identifies the parameter being specified; for example, TYPE=X or LENGTH=(X,X,X). The TYPE entry differentiates between fixed- and variable-length records; the LENGTH entry specifies the length of the input records and the length of the output records. The entries enclosed in parentheses and the entry following TYPE= are values. A value can be a numeric entry, or it can be an alphabetic entry that specifies one of two or more alternatives. Some field definers require no complementary values. The field definer PRINT, for example, is a complete parameter.

In Figure 4, the statement definer is RECORD; the field definers are TYPE and LENGTH; and the values are F (for fixed length), 200 (length of the input records), and 100 (length of the output records). An alternative entry for the value F is V, for variable length. Any numeric values that observe the program's input/output specifications could be used as complement values for the LENGTH field definer.


IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
80-COLUMN CARD PUNCH LAYOUT

NAME	PROJECT NO.	PROJECT I D.	PROJECT NAME	CARD PUNCH PRINT Yes ☐ No ☐	VERIFY Yes ☐ No ☐

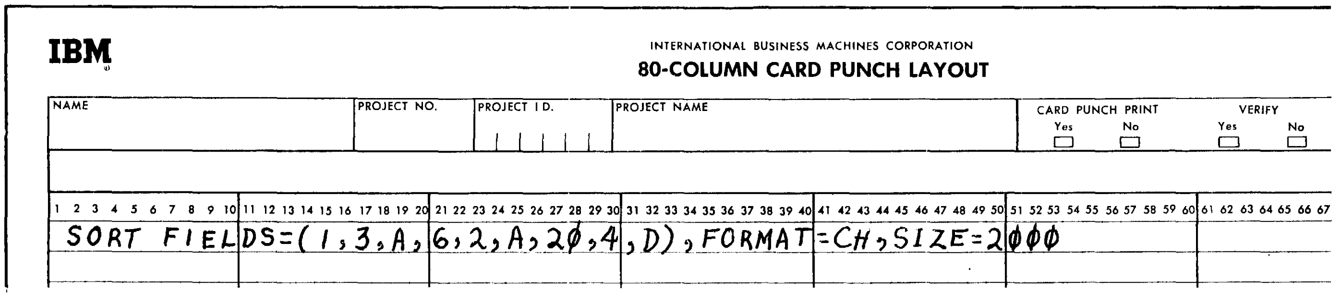
1 2 3 4 5 6 7 8 9 10	11 12 13 14 15 16 17 18 19 20	21 22 23 24 25 26 27 28 29 30	31 32 33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48 49 50	51 52 53 54 55 56 57 58 59 60	61 62 63 64 65 66 67
RECORD TY	PE=F, LENGT	H=(2ØØ,,1Ø	Ø)			

Figure 4. RECORD Statement

Entries in control-statement cards are governed by the following stipulations:

1. The first entry in a control-statement card must be a statement definer, except in a continuation card. The statement definer must be contained between column 2 and column 15, and it must be followed by at least one blank column.
2. Field definers can appear in any order. Each may be followed by one or a list of associated values and a comma. The last field is followed by a blank.
3. Commas, equal signs, right parentheses, and left parentheses must be used only as field delimiters. They cannot be used as values in definers. A comma or blank indicates the termination of a field definer and/or a value. The last field definer or value in a statement must be followed by a blank.
4. A value entry can contain a maximum of five alphameric characters. If the entry exceeds this maximum, an error message is generated. Two exceptions exist: the SIZE value in the SORT statement and the MESSAGES value in the OPTION statement can exceed the maximum.
5. The card deck containing the sort/merge control statements for a given sort or merge application must not exceed 25 cards.

If a series of equal values is enclosed in parentheses, all of the values after the first can be omitted (the parentheses can


IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

80-COLUMN CARD PUNCH LAYOUT

NAME	PROJECT NO.	PROJECT I D.	PROJECT NAME	CARD PUNCH PRINT Yes ☐ No ☐	VERIFY Yes ☐ No ☐

1 2 3 4 5 6 7 8 9 10	11 12 13 14 15 16 17 18 19 20	21 22 23 24 25 26 27 28 29 30	31 32 33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48 49 50	51 52 53 54 55 56 57 58 59 60	61 62 63 64 65 66 67
SORT FIEL	DS=(1,3,A,	6,2,A,2Ø,4	,D),FORMAT	=CH,SIZE=2	ØØØ	

Figure 5. SORT Statement

also be omitted) and the sort program will assume that they are equal to the first. For example, a RECORD statement might contain a single LENGTH value if the input records and the output records are to be the same length. Instead of entering LENGTH= (80,,80) the user might simply enter LENGTH= (80), or even LENGTH=80. The program will assume a second value of 80. If one or more internal values in the series are equivalent to the preceding value, the internal value can be omitted. Each omission must be indicated by a comma.

The following section describes the acceptable control statements, field definers, and values. See Figure 12 at the end of this section for a graphic summary of these entries.

SORT STATEMENT

This statement is required for sorting.

Statement definer: SORT

Field definers with value formats:

FIELDS=(<u>P1</u>,<u>M1</u>,<u>S1</u>,<u>P2</u>,<u>M2</u>,<u>S2</u>,...<u>P12</u>,<u>M12</u>,<u>S12</u>)

FORMAT=<u>XX</u>

SIZE=<u>N</u>

<u>Example</u>: Figure 5. SORT Statement

The FIELDS entry is used to specify the byte location, the length, and the sequence for each control-data field.

P<u>n</u> = the position, within the record, of the high-order byte of the control-data field. If a key field has been specified (fixed-length unblocked records only), its length must be taken into consideration in the P1 value (see <u>OPTION Statement</u>). If, for example, a key length of two bytes is specified and the high-order byte of control-data field 1 is the first byte in the data portion of the record, the value substituted for P1 is 3.

For variable-length records, the value substituted for P1 must take into consideration the RL field.

M<u>n</u> = the length in bytes of the control-data field. The minimum length is 1 byte and the maximum length is 256 bytes.

S<u>n</u> = the sequence into which the control-data field will be sorted. Punch A for ascending and D for descending.

A maximum of 12 control-data fields can be specified with this format. However, the sum of the values specified in M1 through M12 must not exceed 256 bytes. In the example, the major control field (control-data field 1) begins at the first byte of the record, is 3 bytes long, and will be sorted into ascending sequence. Control-data field 2 begins at the sixth byte in the record, is 2 bytes long, and will be sorted into ascending sequence. Control-data field 3 begins at the twentieth byte in the record, is 4 bytes long, and will be sorted into descending sequence.

The FORMAT entry specifies the format of the data in all the control-data fields. Punch one of the following two-character abbreviations.

Character	CH
Zoned Decimal	ZD
Packed Decimal	PD
Fixed Point	FI
Binary	BI
Floating Point	FL

The SIZE entry specifies the number of records in the input file. Punch the exact number or the estimate of the maximum number of input records. The sort/merge program validates the specified work area and operates at greater efficiency even if the SIZE entry is an estimate.

MERGE STATEMENT

This statement is required for a merge-only operation.

Statement definer: MERGE

Field definers with value formats:

FIELDS=(P1,M1,S1,P2,M2,S2,...,P12,M12,S12)

FORMAT=XX

ORDER=M

Example: Figure 6. MERGE Statement

The MERGE statement performs the same basic function as the SORT statement, except that it is used for program runs that will merge records instead of sorting them. The corresponding entries specify the same information in both the SORT and MERGE statements.

The ORDER entry specifies the number of input files to a merge-only operation. Punch a value from 1 to 4 as the definer complement. If a value of 1 is punched, a sequence check will be performed. A reblocking operation also can be performed.

RECORD STATEMENT

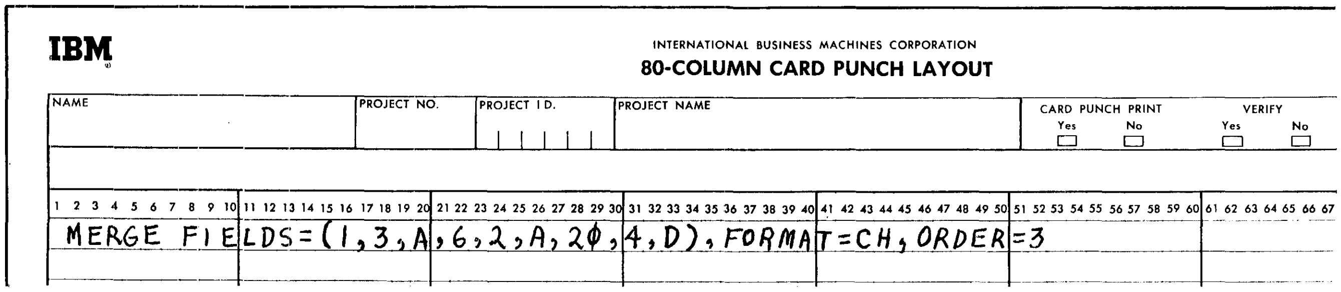
This statement is required.

Statement definer: RECORD

Field definers with value formats:

TYPE=X

LENGTH=(L1, L2, L3) -- for fixed-length records



IBM

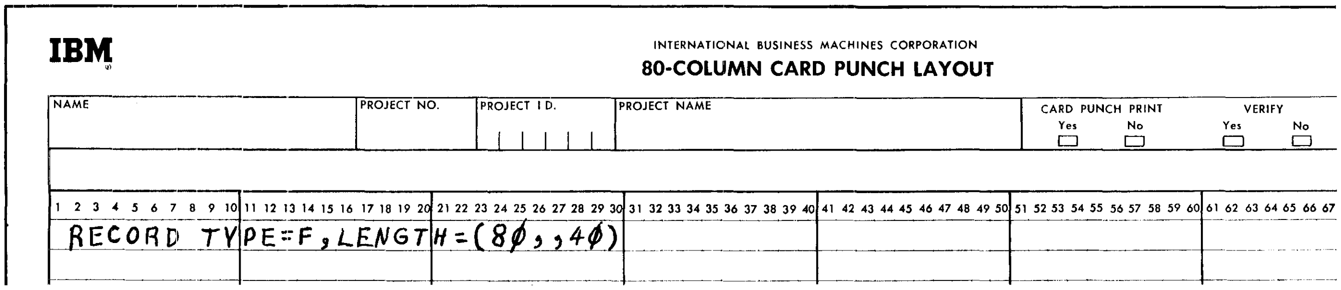
INTERNATIONAL BUSINESS MACHINES CORPORATION

80-COLUMN CARD PUNCH LAYOUT

NAME	PROJECT NO.	PROJECT I D.	PROJECT NAME	CARD PUNCH PRINT Yes ☐ No ☐	VERIFY Yes ☐ No ☐

1 2 3 4 5 6 7 8 9 10	11 12 13 14 15 16 17 18 19 20	21 22 23 24 25 26 27 28 29 30	31 32 33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48 49 50	51 52 53 54 55 56 57 58 59 60	61 62 63 64 65 66 67
MERGE FIE	LDS=(1,3,A	,6,2,A,2Ø,	4,D),FORMA	T=CH,ORDER	=3	

Figure 6. MERGE Statement



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

80-COLUMN CARD PUNCH LAYOUT

NAME	PROJECT NO.	PROJECT I D.	PROJECT NAME	CARD PUNCH PRINT Yes ☐ No ☐	VERIFY Yes ☐ No ☐

1 2 3 4 5 6 7 8 9 10	11 12 13 14 15 16 17 18 19 20	21 22 23 24 25 26 27 28 29 30	31 32 33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48 49 50	51 52 53 54 55 56 57 58 59 60	61 62 63 64 65 66 67
RECORD TY	PE=F,LENGT	H=(8Ø,,4Ø)				

Figure 7. RECORD Statement

LENGTH=(L1,L2,L3,L4)--for variable-length records

Example: Figure 7. RECORD Statement

This statement is required to define the type of record and the length of the records at two points in the program. The TYPE entry is used to differentiate between fixed-length and variable-length records. For fixed-length records, punch an F as the value; for variable-length records, punch a V as the value. In the example, the records are fixed length.

The LENGTH entry for fixed-length records specifies:

L1(required)--the number of bytes in a single input record.

L2--not used by the sort/merge program. Punch a comma as a substitute value.

L3--the number of bytes in a single output record. This entry is significant only if records have been shortened during phase 3 or phase 4. L3 cannot be greater than L1.

The record length specified for fixed-length records must include the KEY length if KEY has been specified in the OPTION statement.

The LENGTH entry for variable-length records specifies:

L1(required)--the maximum number of bytes in a single input record.

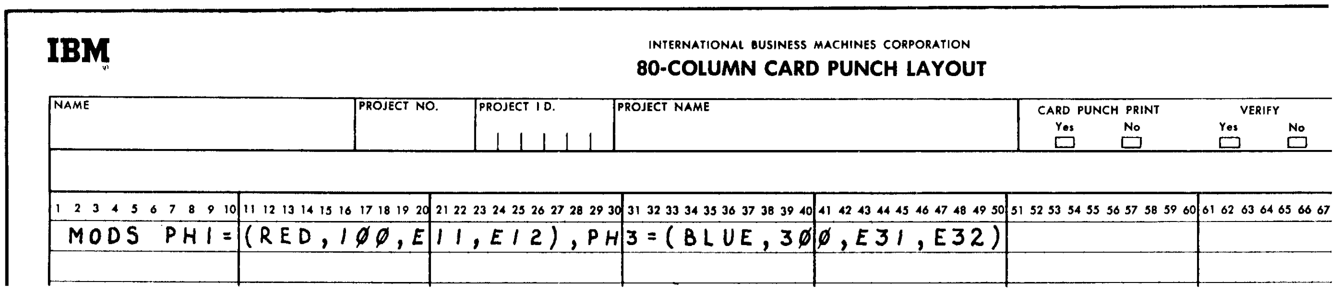
L2--not used by the sort/merge program. Punch a comma as a substitute value.

L3--the maximum number of bytes in a single output record. L3 cannot be greater than L1.

L4(required)--the minimum number of bytes in a single input record.

The record lengths specified for variable-length records must include the four-byte record-length field at the beginning of each record (see Figure 3).

In the example, the input records will be 80 bytes long, and the output records will be 40 bytes long.


IBM

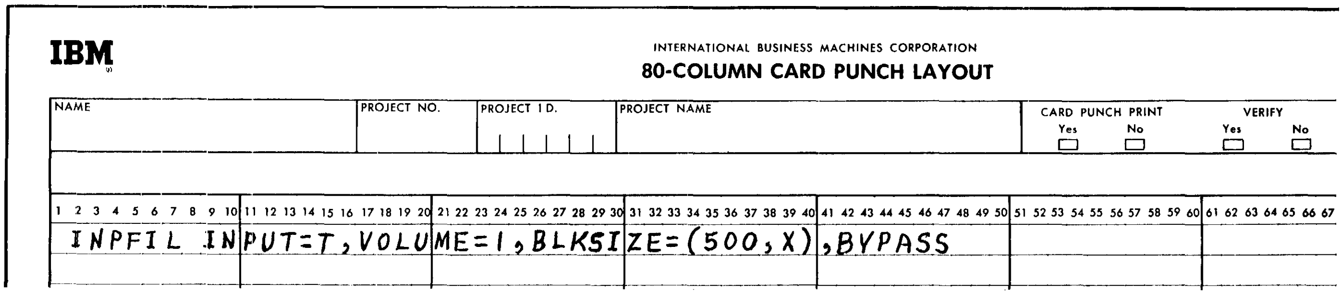
INTERNATIONAL BUSINESS MACHINES CORPORATION

80-COLUMN CARD PUNCH LAYOUT

NAME	PROJECT NO.	PROJECT I D.	PROJECT NAME	CARD PUNCH PRINT Yes ☐ No ☐	VERIFY Yes ☐ No ☐

1 2 3 4 5 6 7 8 9 10	11 12 13 14 15 16 17 18 19 20	21 22 23 24 25 26 27 28 29 30	31 32 33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48 49 50	51 52 53 54 55 56 57 58 59 60	61 62 63 64 65 66 67
MODS PH1=	(RED,1ØØ,E	11,E12),PH	3=(BLUE,3Ø	Ø,E31,E32)		

Figure 8. MODS Statement


IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

80-COLUMN CARD PUNCH LAYOUT

NAME	PROJECT NO.	PROJECT I D.	PROJECT NAME	CARD PUNCH PRINT Yes ☐ No ☐ VERIFY Yes ☐ No ☐

1 2 3 4 5 6 7 8 9 10	11 12 13 14 15 16 17 18 19 20	21 22 23 24 25 26 27 28 29 30	31 32 33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48 49 50	51 52 53 54 55 56 57 58 59 60	61 62 63 64 65 66 67
INPFIL IN	PUT=T,VOLU	ME=1,BLKSI	ZE=(500,X)	,BYPASS		

Figure 9. INPFIL Statement

If L3 of the LENGTH entry is not included, the program assumes the records will not be shortened during phase 3 or phase 4.

MODS STATEMENT

This statement is required if user-written subroutines will be added to the program.

Statement definer: MODS

Field definers with value formats:

PH1=(NAME,LENGTH,E11,E12,E13)

PH3=(NAME,LENGTH,E31,E32)

PH4=(NAME,LENGTH,E41,E42,E43,E44)

Example: Figure 8. MODS Statement

This statement is required to specify the amount of main storage required to store the user-written routines for each phase. It also specifies the exit(s) used to branch out of the mainline program to the user's routine.

The PHn entry specifies the phase in which the exit occurs. The 1 specifies phase 1; 3 specifies phase 3; and 4 specifies phase 4.

The NAME entry for any phase is the name used to catalog the user's subroutines for that phase. The subroutines for each phase must be designated with a unique catalog name.

Punch the actual name (five bytes or less) of the subroutines in the name location of the control statement. A suggested naming procedure for user routines is given in the section User Prepared Routines.

The LENGTH value is the total number of bytes required for the user's subroutines and the associated branch table (see User-Prepared Routines) plus any adjustment required to observe a double-word boundary. This value is determined by subtracting the origin address specified in the PHASE card (used for loading the subroutines into the core image library) from the STORAGE value specified in the OPTION statement.

The size of the user routine is dependent upon the restriction it places upon the maximum Input/Output block and record lengths (see Appendix C). The user routine cannot be such that it forces the maximum allowable I/O block length or record length under 400 bytes for fixed-length records or for ADDROUT (fixed-length or variable-length records) or under 300 bytes for a variable-length record sort. For a merge-only run, the user routine cannot be such that it forces the maximum allowable I/O block length or record length under 300 bytes.

INPFIL STATEMENT

This statement is required.

Statement definer: INPFIL

Field definers with value formats:

INPUT=X

VOLUME=N or (N,N,N,N)

BLKSIZE=(N,X)

OPEN=N

CLOSE=N

BYPASS

Example: Figure 9. INPFIL Statement

This statement defines the input file to be sorted or the files to be merged, and it specifies the procedure that will be followed when the input file is opened or closed.

The INPUT entry specifies the input medium. The values are C for card input, D for disk input, or T for tape input. If card input is specified, the program assumes the input file will be read from SYSIPT. An END card must be the last card of the card-input file. This card must contain a slash mark(/) in column 1 and an asterisk (*) in column 2. Columns 3-80 must be left blank.

The VOLUME entry (tape input only) specifies the number of tape volumes in the input file. Only one entry is needed for a sort operation. For merge-only operation, as many as four values can be entered. The first value corresponds to FILEA, the second value to FILEB, the third value to FILEC, and the fourth value to FILED. The number of volumes should be punched as the value. If no value is specified for this entry and unlabeled files or files with nonstandard labels are being processed, the sort or merge program assumes that each input file is contained in a single volume. If no value(s) is specified and <u>standard labeled</u> files are being processed, the sort or merge program will continue processing until an end-of-file record is read. For standard labeled files, the EOF indication or the specified volume entry takes precedence over the other, depending on which one is encountered first.

The BLKSIZE entry specifies the number of bytes in an input block of records and the type of blocking. The first value is the number of bytes per input block for fixed-length blocked records, or the number of bytes per input block (including key length, if KEYLEN is specified) for fixed-length unblocked records, or the maximum number of bytes per block (including the block-length field and record-length fields) for variable-length blocks. The second value is X for fixed-length blocks. If a second value does not appear, the program assumes variable-length blocks. X may be specified if the file contains fixed-length blocks with the last block in the entire file a short block.

The OPEN entry (for tape input only) specifies whether the first volume of tape input must be rewound before it is read. The values are RWD if the tape must be rewound, or NORWD if it will not be rewound. If the OPEN entry is omitted, the first volume will be opened with a rewind. Whether or not subsequent volumes of a multi-volume file will be opened with a rewind is determined by the CLOSE entry. If the CLOSE entry specifies NORWD, the subsequent volumes will be opened as is. If the CLOSE entry specifies RWD or UNLD, the subsequent volumes will be opened with a rewind.

The CLOSE entry (for tape input only) specifies whether the last volume of the tape input file must be rewound, rewound and unloaded, or left as is. The values are RWD for rewind only, UNLD for rewind and unload, or NORWD if the tape will be left as is. If the CLOSE entry is omitted, the last volume and all previous volumes will be rewound and unloaded.

The BYPASS entry for sort specifies that permanently unreadable input data blocks from tape will be indicated and bypassed without being processed. If a permanently unreadable input data block occurs with disk input, the sort will be terminated. The BYPASS entry for the merge-only operation specifies that permanently unreadable input data blocks from tape or disk will be indicated and bypassed without being processed. If this entry is omitted, the system will enter the wait state when an unreadable block is detected. If EXIT 13 or EXIT 44 is used to examine the unreadable data block, the block will be bypassed, whether BYPASS was specified or not, after the exit has been taken.

OUTFIL STATEMENT

This statement is required.

Statement definer: OUTFIL

Field definers with value formats:

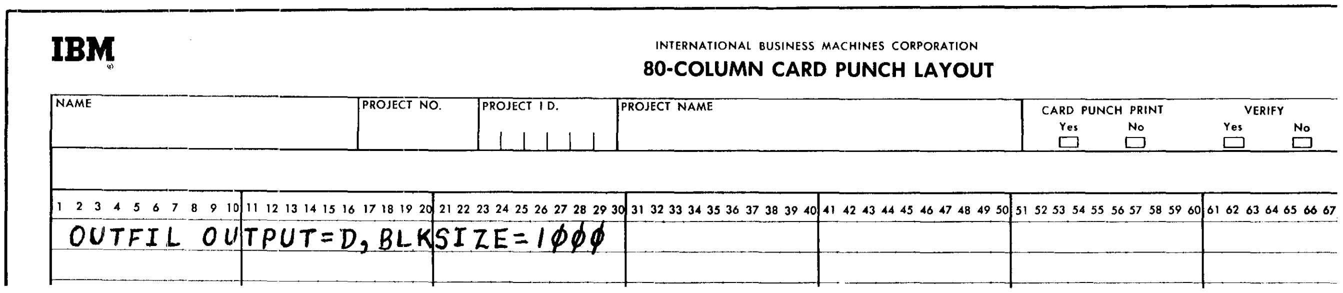
OUTPUT=<u>X</u>

BLKSIZE=<u>N</u>

OPEN=<u>N</u>

CLOSE=<u>N</u>

Example: Figure 10. OUTFIL Statement


IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

80-COLUMN CARD PUNCH LAYOUT

NAME	PROJECT NO.	PROJECT I D.	PROJECT NAME	CARD PUNCH PRINT Yes ☐ No ☐	VERIFY Yes ☐ No ☐

1 2 3 4 5 6 7 8 9 10	11 12 13 14 15 16 17 18 19 20	21 22 23 24 25 26 27 28 29 30	31 32 33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48 49 50	51 52 53 54 55 56 57 58 59 60	61 62 63 64 65 66 67
OUTFIL OU	TPUT=D,BLK	SIZE=1∅∅∅				

Figure 10. OUTFIL Statement

The OUTFIL statement defines the output file and specifies the procedure that will be followed when the output file is opened or closed.

The OUTPUT entry specifies the output medium. The values are D for disk output, W for disk output if a portion of the output overlaps the work area, or T for tape output.

The BLKSIZE entry specifies the number of bytes per output block for blocked fixed- length records, or the number of bytes per record (including key length if KEYLEN is specified) for unblocked fixed-length records, or the maximum number of bytes in a single block (including the block-length field or record-length fields) for variable-length records. For unblocked output with fixed-length records, specify a BLKSIZE value equal to the output record length. For output with variable-length records, the user must specify a BLKSIZE value at least equal to the maximum input record length plus four bytes for the block-length field.

The OPEN entry (for tape output only) specifies whether the first volume of a tape output file must be rewound before the final output file is written. The values are RWD if the tape will be rewound, or NORWD if it will not be rewound. If the OPEN entry is omitted, the sort program assumes the tape must be rewound. Subsequent volumes will be rewound.

The CLOSE entry (for tape output only) specifies whether the last volume of a tape output file must be rewound, rewound and unloaded, or left as is. The values are RWD for rewind only, UNLD for rewind and unload, or NORWD if the tape will be left as is. If the CLOSE entry is omitted, the last volume will automatically rewind but not unload. Previous volumes will be rewound and unloaded.

OPTION STATEMENT

This statement is required.

Statement definer: OPTION

Field definers with value formats:

RESTART

MESSAGES=N

PRINT

STORAGE=N

VERIFY

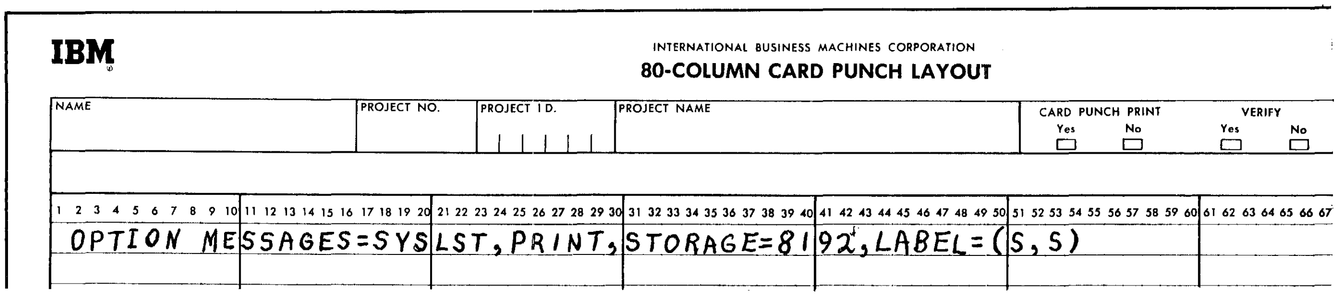
KEYLEN=X

ADDROUT

LABEL=(I,O)

NOTPMK

Example: Figure 11. OPTION Statement


IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

80-COLUMN CARD PUNCH LAYOUT

NAME	PROJECT NO.	PROJECT I.D.	PROJECT NAME	CARD PUNCH PRINT Yes ☐ No ☐	VERIFY Yes ☐ No ☐

1 2 3 4 5 6 7 8 9 10	11 12 13 14 15 16 17 18 19 20	21 22 23 24 25 26 27 28 29 30	31 32 33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48 49 50	51 52 53 54 55 56 57 58 59 60	61 62 63 64 65 66 67
OPTION ME	SSAGES=SYS	LST,PRINT,	STORAGE=81	92,LABEL=(	S,S)	

Figure 11. OPTION Statement

This statement defines several of the sort/merge options. The RESTART entry is included only if the sort run has been interrupted and is being restarted (see Checkpoint, Interrupt, and Restart). All control statements included in the initial sort run must be included for a restart run.

The required value included in the MESSAGES entry is the symbolic name of the device that will be used to print diagnostic messages during the assignment phase. This device must be either SYSLOG or SYSLST.

The PRINT entry specifies that the control card, record counts, and end-of-phase messages will be printed by the device specified in the MESSAGES entry.

The STORAGE entry specifies the total storage that may be used by the sort/merge program and user modifications. The storage above this limit will not be disturbed by the sort program. The value specified must be equal to or greater than 8192. If this value is not specified, the storage value specified in the CONFG statement (job-control statement) will be assumed.

The VERIFY entry specifies that the disk output file will be program-verified, that is, the output file has been written correctly. If this entry is omitted, the verification will not be performed. It should be noted that the verify option considerably increases sorting time.

The KEYLEN entry (disk files only) specifies the length of the KEY that is associated with each input record. Punch the number of bytes in a key as the value for the entry. If a key length is specified, each record must have an associated key, and each key must be of the specified length. The sort/merge program will process records that are equal in length to the specified data-record length plus the specified key length. A KEYLEN entry can be specified for unblocked, fixed-length records only.

The output file must be unblocked and will consist of the key plus the data record. If an input file consists of unblocked, fixed-length records with keys, but the user does not wish to specify the key to the sort, the KEYLEN entry should not be specified and the length of the key should be omitted from the input record size. The key, therefore, will not appear in the sorted output file. If blocked, fixed-length input records with keys are to be sorted, KEYLEN should not be specified and the sort/merge program will drop the key and process only the actual record.

The ADDROUT entry specifies that the final output from the sort will be a string of disk addresses in binary format. The output file can be written on disk or tape. If the output file is written on tape, the block must consist of at least two records. If the last output block is not a full block, the last valid record will be followed by an eight-character record of blanks. The data records will not be written out in the output file; addresses that permit the records to be retrieved in sorted sequence will be written out. The ADDROUT entry can be specified only if the input file is read from disk and the entire input area is online throughout the sort. Address output will have the following format:

MCCHHRBB

Address length = 8 bytes

M = number of the pack (0-254) on which the sort input record is located. All packs for a file must be numbered consecutively starting with 0. That is, the first pack of sort input must be number 0, etc. This number relates to a numbered symbolic unit (SYS000-SYS254). Two or more symbolic units for a file must be numbered consecutively, but the numbering may start with any SYSnnn number. See track referencing in the Assembler manual listed in the preface of this publication.

CC = cylinder number

HH = read/write head

R = number of the data record on the track

BB = zero for unblocked, fixed-length records, or the displacement, in bytes, of the record within the block. Four for unblocked, variable-length records or the displacement, in bytes, of the record within the block.

If disk-address output is specified in the OPTION statement, the total length of all control fields plus 8 bytes (CF +8) cannot exceed the input record length.

When EXIT 12 occurs, and disk address output has been specified, the records being processed will contain the 8-character disk address followed by the control data in the specified sequence. When EXIT 32 occurs, the records being processed will have the format of an eight character disk address followed by the specified control data. User translation of output data cannot be performed through EXIT 32 if ADDROUT is specified.

The specified output block length, if ADDROUT is specified, must be a multiple of 8 and must observe the limits specified in the section Record Lengths and Input/Output Block Lengths.

The KEYLEN entry can be specified with ADDROUT, and records can then be sorted on the key field, but the key will not appear in the final disk-address output file.

The LABEL entry specifies the type of labels on the input and output files. The first value is for input labels, and the second value is for output labels. Substitute one of the following values for I and one for O:

N = Nonstandard labels
S = Standard labels with no user labels
U = Unlabeled files
A = Standard labels with user labels

Note: For card input, specify U for unlabeled input file.

The NOTPMK entry indicates that a tape mark will not be written before the first data record in each volume of the output file. This field definer should be included only when an unlabeled output file or an output file with nonstandard labels will be written. If NOTPMK is specified and standard output labels are specified, an error message will be printed, and the program will ignore the NOTPMK definer (the tape mark will be written).

END STATEMENT

This statement is required.

Statement definer: END

Field definers with value formats: none

The END statement indicates the end of the control-statement deck.

Statement Definer	Status	Field Definer (Status) and Value Format
SORT	Required for sort runs	FIELDS (Required) = (P1, M1, S1) FORMAT (Required)= XX SIZE (Required)= N
MERGE	Required for merge runs	FIELDS (Required) = (P1, M1, S1) FORMAT (Required) = XX ORDER (Required)= M
RECORD	Required	TYPE (Required) = X LENGTH (Required)= (L1, L2, L3) = (L1, L2, L3, L4)

MODS	Required if user-written subroutines will be added	Ph1=(NAME, LENGTH,E11, E12, E13)
		Ph3= (NAME, LENGTH, E31, E32)
		Ph4= (NAME, LENGTH, E41, E42, E43, E44)

INPFIL	Required	INPUT (Required) = X VOLUME (Optional) = N BLKSIZE (Required)= (N,X) OPEN (Optional) = N CLOSE (Optional) = N BYPASS (Optional)
OUTFIL	Required	OUTPUT (Required) = X BLKSIZE (Required) = N OPEN (Optional) = N CLOSE (Optional) = N
OPTION	Required	RESTART (Optional) MESSAGES (Required) = N PRINT (Optional) STORAGE (Optional) = N KEYLEN (Optional) = X ADDROUT (Optional) LABEL (Required) = (I,O) NOTPMK (Optional) VERIFY (Optional)
END	Required	

Figure 12. Summary of Sort/Merge Control Statements

At least one storage area, the work area, must be reserved on disk for the use of the sort program. If the output file will be written on disk, an output area must be reserved also. These two areas must be calculated by the user, and their limits must be defined to the sort program with extent cards. The formulas discussed in the following section should be used to determine the extent of these two areas. The results obtained from these formulas will provide the user with maximum utilization of his available disk storage.

WORK AREA

The total amount of disk required for the work area is twice the number of tracks required to store the input file, after it has been reblocked for sorting, plus four tracks required to store phase-to-phase information and checkpoint information. The total work area may be any number of tracks up to a maximum of four full packs. The work area need not be a single continuous extent. It can be divided into four areas which, when taken as a whole, are equivalent to the required total. Each area must be defined with an extent card (see Job-Control Statements). The last extent card specifying the work area must include at least four tracks.

The following series of calculations will produce two work-area sizes: the maximum and the minimum. These calculations are valid for fixed-length records only. To calculate the work area for variable-length records, see Appendix A. To calculate the work area for disk-address output, with fixed-length or variable-length records, see Appendix B.

If the calculated maximum work area and the maximum input block length are specified at execution time, the assignment phase of the sort program will select an order of merge and sort blocksize that will minimize the sorting time. If the calculated minimum work area is specified, the assignment phase is limited in its selection of the order of merge and sort blocksize, reducing the efficiency of the sort program.

The calculation of the sort blocking factors is arranged in a sequence of steps. Each step except the first should be carried out for each possible order of merge (M). If a main-storage value less than 16,384 is specified, M ranges from 2 through 4; if a main-storage value equal to or greater than 16,384 is specified, M ranges from 2 through 7. It is suggested that the calculated values be arranged into a table, as in Figure 13.

Before performing the following calculations, determine the maximum input and output block lengths (BIL and BOL) specified in the section Record Length and Input/ Output Block Lengths. The symbols used in the following formulas are defined in Symbol Definition.

Step 1

Calculate a value Ph1B1 by solving the two following equations.

$$\text{Ph1B1} = \left\lfloor \frac{\text{ST-Sup-Ph1-UA1}}{2*L_1+8} \right\rfloor$$

and

$$\text{Ph1B1} = \left\lfloor \frac{3624}{L_1} \right\rfloor$$

The smaller of the two resultant values is the effective Ph1B1 and should be retained for further consideration. Ph1B1 is the maximum number of records that can be contained in a single phase-1 sort block (B) if the number of records in a phase-1 sort block is some multiple of the user's specified input blocking factor (BI). Because the number of records in a phase-2 and phase - 3 sort block is unknown at this point, two additional equations must be solved to determine the maximum number of records that can be contained in a phase-1 sort block if it cannot contain a multiple of the user's input blocking factor.

Note: For variable-length blocks of fixed-length records, eliminate the calculation of Ph1B1 and its consideration throughout the work-area calculations.

$$\text{Ph1B2} = \left\lfloor \frac{\text{ST-Sup-Ph1-UA1-BIL}+L_1}{2*L_1+8} \right\rfloor$$

and

$$\text{Ph1B2} = \left\lfloor \frac{3624}{L_1} \right\rfloor$$

The smaller of the two resultant values is the effective Ph1B2 and should be retained for further consideration.

M	Max Block Length	Max Recs/Block	Recs/Block	Bytes/Block	Blocks/Track	Bytes/Track	Alt Blocks/Track	Alt Recs/Block	Alt Bytes/Block	Alt Bytes/Track
2										
3										
4										
5										
6										
7										

Figure 13. Sample Calculation Chart

Step 2

Calculate Maximum Block Length (Max BL), the maximum number of bytes in a single phase-2 or phase-3 block, with the following formulas:

$$\text{Phase-2 Max BL} = \left\lfloor \frac{\text{ST-Sup-Ph2}}{M + 1} \right\rfloor$$

$$\text{Phase-3 Max BL} = \left\lfloor \frac{\text{ST-Sup-Ph3-8-UA3-BOL-}L_1+L_3}{M} \right\rfloor$$

The smaller of the resultant values must be used as Max BL.

If Max BL exceeds 3624, substitute the value 3624 for it. The value Max BL should be charted as in Figure 13.

Step 3

Calculate Maximum Records per Block (Max Recs/Blk), the maximum number of records that can be contained in a phase-2 and phase-3 block, with the following formula:

$$\text{Max Recs/Blk} = \left\lfloor \frac{\text{Max BL}}{L_1} \right\rfloor$$

Chart this value as in Figure 13.

If the Max Recs/Blk value is $\leq$ Ph1B2, it must be used as the Recs/Blk value, also, and charted as such. If the Max Recs/Blk value is greater than Ph1B2, calculate the largest multiple of BI that is $\leq$ the effective Ph1B1.

$$\left\lfloor \frac{\text{Ph1B1}}{\text{BI}} \right\rfloor * \text{BI}$$

Compare the resultant value with the result of:

$$\left\lfloor \frac{\text{Max Recs/Blk}}{\text{BI}} \right\rfloor * \text{BI}$$

Compare the smaller of these two values

with Ph1B2. The greater of these two values should be charted as Recs/Blk.

Step 4

Calculate the number of Bytes per Block (Bytes/Blk) using the value of Recs/Blk determined by the last comparison.

Bytes/Blk = Recs/Blk * L_1

Chart the resultant value as in Figure 13.

Step 5

Consult Figure 14 to determine the number of blocks that will fit on a single disk track (Blks/Track). In Figure 14, column 1 contains the number of blocks that will fit on a track if the block length is no greater than the corresponding value in column 2. (Figure 14 should not be used for input/output file considerations.) For example, 5 blocks will fit on a track if the Bytes/Blk ≤ 649.

Chart the number of Blks/Trk.

Next, calculate the number of bytes per tracks (Bytes/Trk) based upon the Blks/Trk just determined. Use the following relation.

Bytes/Trk = Blks/Trk *Bytes/Blk = Blks/Trk * Recs/Blk * L_1

Chart the number of Bytes/Track.

Step 6

It may be possible to improve the utilization of available disk storage by decreasing the number of Recs/Blk sufficiently to increase the number of Blks/Trk by 1. Refer to Figure 14 to determine the maximum number of Bytes/Blk if the value Blks/Trk is increased by 1. Chart the increased Blks/ Trk value as Alt Blks/Trk, and on the basis of this figure, calculate a decreased number of Recs/Blk (Alt Recs/Blk) and perform the following checks. If Alt Recs/Blk is ≤ Ph1B2, chart it as the Alt Recs/Blk. If Alt Recs/Blk is > Ph1B2 and > BI also, make Alt Recs/Blk a multiple of BI with the following formula:

$$\left\lfloor \frac{\text{Alt Recs/Blk}}{\text{BI}} \right\rfloor * \text{BI}$$

Chart the multiple as Alt Recs/Blk. If BI is > Alt Recs/Blk, make Alt Recs/Blk a submultiple of BI and chart the submultiple as Alt Recs/Trk.

Column 1 Blocks/Track	Column 2 Maximum Block Length
1	3624
2	1738
3	1129
4	829
5	649
6	530
7	445
8	382
9	333
10	293
11	261
12	234
13	211
14	192
15	175
16	160
17	147
18	136
19	126
20	116
21	108
22	100
23	93
24	87
25	81
26	76
27	71
28	66
29	62
30	58
31	54
32	50
33	47
34	44
35	41
36	38
37	35
38	33
39	31
40	28
41	26
42	24
43	22
44	20
45	19
46	17
47	15
48	14
49	12
50	11

Figure 14. BLK/TRK WITH Maximum Block Length

Calculate the decreased Bytes/Blk (Alt Bytes/Blk) with the following equation:

Alt Bytes/Blk = Alt Recs/Blk * L_1

Chart the Alt Bytes/Blk.

Next, calculate the total number of bytes (Alt Bytes/Trk) that can be contained on a track if the alternate blocking configuration is used. Use this equation for the calculation:

Alt Bytes/Trk=
Alt Blks/Trk * Alt Recs/Blk * L_1

Now, compare the value of Bytes/Trk to the value of Alt Bytes/Trk. If Alt Bytes/Trk is ≤ the initial Bytes/Trk, use of the alternate blocking configuration will not improve the utilization of available disk storage. However, if Alt Bytes/Trk is > the initial Bytes/Trk, use of the alternate configuration will improve disk utilization.

The calculations required for a single order of merge have been completed. This process (steps 2 through 6) should be repeated for each possible order of merge.

Step 7

After the values for each order of merge have been determined, the maximum and minimum work areas can be calculated. From the Bytes/Trk column of the chart just compiled, select the smallest number of Bytes/Trk. This value is the minimum number of Bytes/ Trk (Min Bytes/Trk) that will be calculated by the sort assignment phase. It can be used in the following equation to determine the maximum number of disk tracks (Amax) that can be used advantageously for the work area:

$$\text{Amax} = 2 * \left\lceil \frac{N\ I\ R * L_1}{\text{Min Bytes/Trk}} \right\rceil + 4$$

Next, select the largest of the values that appears in the Bytes/Trk column and the Alt Bytes/Trk column. This value is the maximum number of Bytes/Trk (Max Bytes/Trk) that will be calculated by the sort assignment phase. It can be used in the following equation to determine the minimum number of disk tracks (Amin) that can be used for the work area:

$$\text{Amin} = 2 * \left\lceil \frac{N\ I\ R * L_1}{\text{Max Bytes/Trk}} \right\rceil + 4$$

SYMBOL DEFINITION

*	= multiplied by (2*3=6).
$\lceil\ \rceil$	= rounded high ($\lceil 1.2 \rceil$ =2).
$\lfloor\ \rfloor$	= rounded low ($\lfloor 1.2 \rfloor$=1).
B	= The number of records that can be processed as a single block by the sort program.
BI	= The number of records in each block of input.
BIL	= The length (or maximum length) in bytes, of a single block of input.
BOL	= The length in bytes, of a single block of output.
CF	= The total length of all control fields.
L_1	= The length in bytes, of a single input record.
L_3	= The length in bytes, of a single output record.
M	= The order of merge.
Max BL	= The maximum number of bytes that can be contained in a single block of records in phase 3.
NIR	= The number of records in the input file.

Ph1 = the phase-1 program size. Use one of the following:

1. For fixed-length records: 1848 + 10N + 6M_1 + Data Conversion Routine +Q

   Note: See Symbols following Ph3 formulas.

2. For fixed-length records if ADDROUT is specified:
   1898 + 10N + 6M_1 + Data Conversion Routine +Q

3. For variable-length records:
   1998 + 10N + Data Conversion Routine

4. For variable-length records if ADDROUT is specified: 1898 + 10N + Data Conversion Routine + Q

Ph2 = the phase-2 program size. Use one of the following for a 2-, 3-, or 4-way merge on systems with up to 16,384 positions of main storage specified:

1. For fixed-length records with or without disk-address output:
   910 + 10N + 6M_1

2. For variable-length records:
   1100 + 10N

3. For variable-length records if ADDROUT

is specified:
$910 + 10N + 6M_1$

Use one of the following for a 5-, 6-, or 7-way merge on systems with 16,384 or more positions of main storage specified:

1. For fixed-length records with or without disk-address output:
$1900 + 10N + 6M_1$

2. For variable-length records:
$2100 + 10N$

3. For variable-length records if ADDROUT is specified:
$1900 + 10 N + 6M_1$

Ph3 = the phase-3 program size. Use one of the following for a 2-, 3-, or 4-way merge on systems with up to 16,384 positions of main storage specified:

1. For fixed-length records with or without disk-address output:
$800 + 10N + 6M_1$ + Data Reconversion Routine
2. For variable-length records:
$900 + 10N$ + Data Reconversion Routine

3. For variable-length records if ADDROUT is specified:
$800 + 10N + 6M_1$ + Data Reconversion Routine

Use one of the following for a 5-, 6-, or 7-way merge on systems with 16,384 or more positions of main storage specified:

1. For fixed-length records with or without disk-address output:
$1650 + 10N + 6M_1$ + Data Reconversion Routine

2. For variable-length records:
$1800 + 10N$ + Data Reconversion Routine

3. For variable-length records if ADDROUT is specified:
$1650 + 10N + 6M_1$ + Data Reconversion Routine

Ph4 = the phase-4 program size.

1. $1250 + 10N + 6M_1$ + Data Conversion and Reconversion Routines

<u>Symbols</u>

N = The number of control fields. For Ph2, Ph3, and Ph4 calculation, add 74 to the total if $N > 1$.

$$M_1 = \left\lceil \frac{L_1}{256} \right\rceil$$

Q = 186bytes, which must be added to the program size of Phase 1 if user routines and/or data conversion is specified.

Data Conversion Routine (Phase 1)=

76 bytes for fixed-point data

120 bytes for floating-point data

136 bytes for unpacked-decimal data

180 bytes for packed-decimal data

Data Reconversion Routine (Phase 3) =

76 bytes for fixed-point data

120 bytes for floating-point data

136 bytes for unpacked-decimal data

180 bytes for packed-decimal data

Data Conversion and Reconversion

(Phase 4)=

68 bytes for fixed-point data

128 bytes for floating-point data

198 bytes for unpacked-decimal data

304 bytes for packed-decimal data

ST = The amount of main storage specified. This value should correspond to the value specified in the STORAGE entry of the OPTION statement. If it is not specified, this value should be equal to the machine size specified in the CONFG card (see <u>Control Statements: OPTION Statement</u>).

Sup = The number of bytes of main storage required by the Supervisor and IOCS (see <u>IBM Basic Operating System/360, Programmer's Guide (8K Disk)</u>, C24-3372.)

UA1 = Total size, in bytes, of the phase-1 user routines.

UA3 = Total size, in bytes, of the phase-3 user routines.

UA4 = Total size, in bytes, of the phase-4 user routines.

INPUT OR OUTPUT AREA

Calculate, using the following procedure, the number of tracks required to contain the input or output file.

To determine the number of Blks/Trk, see the Systems Reference Library publication, IBM 2311 Disk Storage Drive, A26-5860.

Next, calculate the number of Recs/Trk

$$\text{Recs/Trk} = (\text{Blks/Trk})\left(\frac{\text{Block Length}}{\text{Record Length}}\right)$$

The number of tracks required for the input or output file can then be determined.

$$\text{Area in Tracks} = \left\lceil \frac{\text{Number of Records}}{\text{Recs/Trk}} \right\rceil$$

The output area can overlap the first half of the work area, but the 4 tracks required by the sort program for phase-to-phase information and checkpoint information must be deducted from the work area before the number of tracks overlapped is determined. The allocation of tracks in the work area is completely dependent upon the extent sequence number in the extent cards defining the work area. For example, if the work area extent card with the extent sequence number 001 specifies a disk area of 10 tracks and the extent card with extent sequence number 002 specifies an area of 74 tracks, then the first half of the work area would consist of the 10 tracks specified in the first extent card and the first 30 tracks specified in the second extent card. The second half of the work area consists of the next 40 tracks specified in the second extent card. The last four tracks specified in the second extent card are for sort purposes. This equation can be used to determine the amount of overlap:

Number of tracks overlapped=

$$\left\lceil \frac{\text{Work Area in Track-4}}{2} \right\rceil$$

If the output area will overlap the work area, an accurate file size must be specified in the SIZE entry of the SORT statement (see Sort/Merge Control Statements: SORT Statement).

MAXIMUM FILE SIZE

The following procedure can be used to calculate the maximum number of records that can be sorted under two conditions: if the maximum amount of disk storage is available, or if the minimum amount of disk storage is available. Calculate these maximums with the following formula:

$$\text{MFS} = \left\lfloor \frac{\text{Available Disk-SA}}{\frac{1}{\text{Input Recs/Trk}} + \frac{1}{Z} + \frac{1}{Y}} \right\rfloor$$

This formula assumes that the output area overlaps approximately the first half of the work area.

MFS = Maximum File Size - the maximum number of records that can be sorted.

Available Disk = The total number of disk tracks available for use during the sorting process for input, work, and output areas.

SA = 4 Tracks - The number of tracks required by the sort program to store phase-to-phase information and checkpoint information.

Input Recs/Trk = The number of input records that can be contained on a single disk track as calculated in the section entitled Input or Output Area.

Z = The number of Recs/Trk in the work area. This number is based upon either the Max Bytes/Trk used to calculate Amin or the Min Bytes/Trk used to calculate Amax (see Step 7 in Work Area).

Calculate Z with one of the following equations:

$$1. Z = \text{Recs/Trk} = \frac{\text{Max Bytes/Trk}}{\text{Record Length}}$$

$$2. Z = \text{Recs/Trk} = \frac{\text{Min Bytes/Trk}}{\text{Record Length}}$$

Equation 1 will produce the maximum number of records per work-area track if the minimum number of tracks is specified. Equation 2 will produce the maximum number of records per work-area track if the maximum number of tracks is specified. The record length in equation 1 and 2 above is either the input record length or CF+8 if ADDROUT is specified.

Y = The smaller of a or b below:

a = The number of output records per track (Recs/Trk) calculated in the section entitled Input or Output Area.

b = The value substituted for Z (Recs/Trk) above.

PLACEMENT OF DISK-STORAGE AREAS

This section describes the suggested relative locations for disk input, output, and work areas.

Although they are not discussed in this section, the system-residence requirements must be taken into account when disk areas are allocated. For information on the system-residence requirements, see IBM Basic Operating System/360, Programmer's Guide (8K Disk), C24-3372.

If the system has one disk drive available for sorting, the areas should be arranged as shown in Figure 15 or as shown in Figure 16.

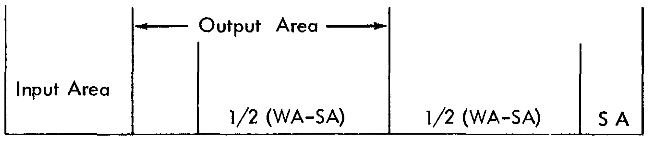


Figure 15. Area Allocation for One Disk Drive

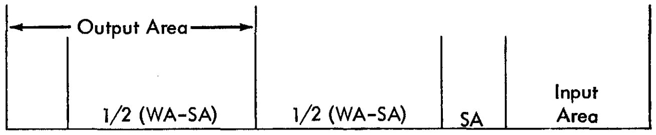


Figure 16. Area Allocation for One Disk Drive

If the output area and the work area will overlap, an accurate file size must be specified in the SIZE entry of the SORT statement.

If the system has two disk drives available for sorting, the output area should be placed on the first drive, overlapping the first half of the work area. The second half of the work area should be placed at the beginning of the second drive. The input area should be placed at the beginning of the first drive or at the end of the second drive (Figure 17).

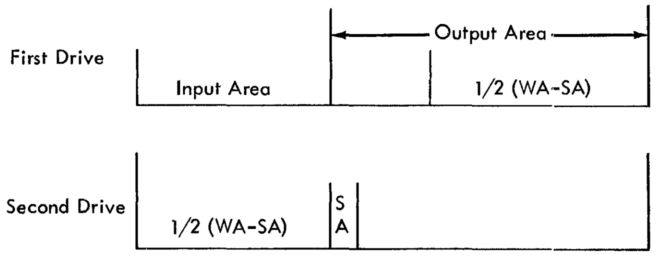


Figure 17. Area Allocation for Two Disk Drives

The output area on the first drive can overlap the work area on the first drive. To do this, an accurate file size must be specified in the SIZE entry of the SORT statement, and the extent card that defines the work area on the first drive must have a lower extent-sequence number than the extent card that defines the work area on the second drive.

If the system has three disk drives available for sorting, the input area should be placed at the beginning of the first drive. The first half of the work area should be placed on the second drive, and the second half of the work area should be placed on the third drive. The output area should immediately follow the input area on the first drive, and if necessary, it should overlap into the work area on the second drive (Figure 18).

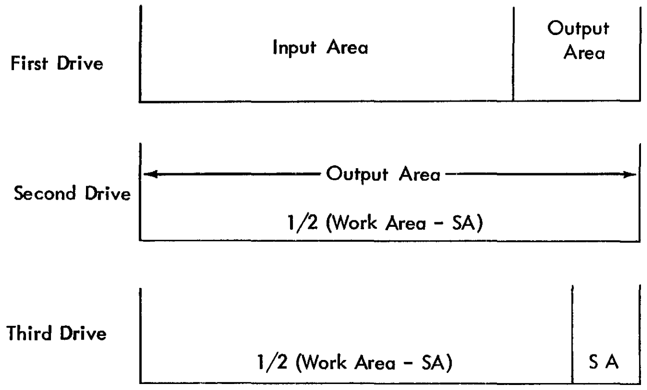


Figure 18. Area Allocation for Three Disk Drives

If the output area overlaps the work area on the second drive, an accurate file size must be specified in the SIZE entry of the SORT statement. Also, the extent card that defines the work area on the second drive must have a lower extent-sequence number than the extent card that defines the work area on the third drive.

If the system has four disk drives available for sorting, the input area should be placed at the beginning of the first drive. The output area should be placed at the beginning of the second drive, or immediately following the end of the input area if the input area exceeds the limit of the first drive. The output area can overlap the work area specified on the third drive. The first half of the work area should be placed on the third drive, and the second half of the work area should be placed on the fourth drive (Figure 19).

If the output area overlaps the work area on the third drive, an accurate file size must be specified in the SIZE entry of the SORT statement. Also, the extent card that defines the work area on the third drive must have a lower extent-sequence number than the extent card that defines the work area on the fourth drive.

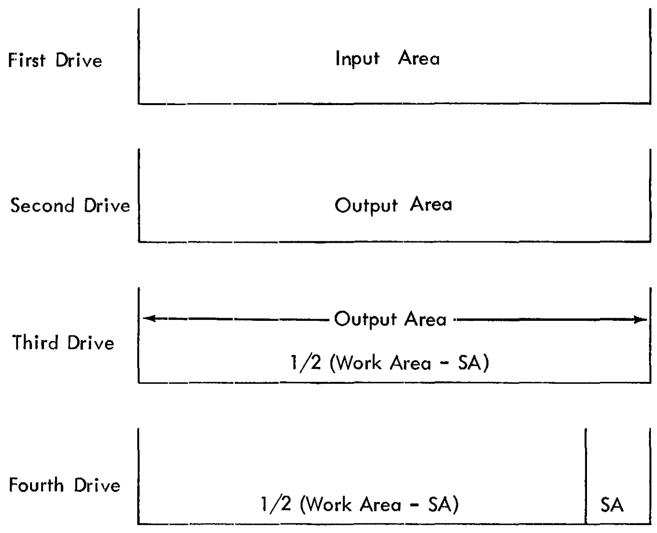


Figure 19. Area Allocation for Four Disk Drives

APPENDIX A: CALCULATION OF THE WORK AREA FOR VARIABLE-LENGTH RECORDS

To calculate the work area for variable-length records, first, calculate three values (MaxBL1, MaxBL2, and MaxBL3) using the following formulas:

$$\text{BL1} = \left\lfloor \frac{\text{ST-Sup-Ph,-UA1-MaxBIL-7}}{\left(2 + \frac{8}{\text{Lmin}}\right)} \right\rfloor$$

$$\text{MaxBL1} = \left\lfloor \frac{\text{BL1}}{\text{Lmax}} \right\rfloor * \text{Lmax}$$

Solving these two equations will produce the maximum acceptable phase-1 block size (MaxBL1). Retain this value for further consideration. For an explanation of the symbols, see <u>Symbol Definition</u>. Lmin (L4) equals the length in bytes of the smallest input record; Lmax (L1) equals the length in bytes of the largest input record; and MaxBIL equals the length, in bytes, of the largest input block. MaxBL1 must be a multiple of Lmax, but it must not exceed 3624.

$$\text{BL2} = \left\lfloor \frac{\text{ST-Sup-Ph2-(Lmax)} \quad \text{(M)}}{\text{M+1}} \right\rfloor$$

$$\text{MaxBL2} = \left\lfloor \frac{\text{BL2}}{\text{Lmax}} \right\rfloor * \text{Lmax}$$

M = 2-3 if less than 16,384 bytes of main storage are specified.

M = 2-6 if 16,384 or more bytes of main storage are specified.

Solving these two equations will produce the maximum acceptable phase-2 block size (MaxBL2). Retain this value for further consideration. MaxBL2 must be a multiple of Lmax, but it must not exceed 3624.

$$\text{BL3} = \left\lfloor \frac{\text{ST-Sup-Ph3-UA3-(Lmax) (M)-BOL-8}}{\text{M}} \right\rfloor$$

$$\text{MaxBL3} = \left\lfloor \frac{\text{BL3}}{\text{Lmax}} \right\rfloor * \text{Lmax}$$

MaxBL3 is the maximum acceptable phase-3 block size. It must be a multiple of Lmax, but it must not exceed 3624.

Compare MaxBL1, MaxBL2, and MaxBL3. The smallest of these three values is the Bytes/Blk value. Next, determine the number of Blks/Trk by consulting Figure 14. Calculate Bytes/Trk with the formula:

Blks/Trk * Bytes/Blk

An alternate Bytes/Blk must be determined next. Begin by increasing the number of Blks/Trk by one (Blks/Trk + 1 = Alt Blks/Trk). Refer to Figure 14 to determine an alternate Bytes/Blk. Round the Alt Bytes/Blk with the following formula:

$$\left\lfloor \frac{\text{Alt Bytes/Blk}}{\text{Lmax}} \right\rfloor * \text{Lmax}$$

This Alt Bytes/Blk value is valid only when it is greater than Lmax.

Calculate Alt Bytes/Trk with the following formula:

Alt Bytes/Trk = Alt Bytes/Blk * Alt Blks/Trk

Compare Bytes/Trk with Alt Bytes/Trk. If the Alt Bytes/Trk is ≤ the initial Bytes/Trk, use of the alternate <u>will not</u> improve the utilization of the available disk area. However, if Alt Bytes/Trk is > the initial Bytes/Trk, use of the alternate <u>will</u> improve utilization.

The number of bytes in a single phase-1 sequence must be calculated. First, determine the amount of storage available during phase 1. Use this equation:

Avail Ph1 Storage =
ST-Sup-Ph1-UA1-MaxBIL-Bytes/Blk-7

<u>Note</u>: From this point on, substitute Alt Bytes/Blk for Bytes/Blk if improved utilization will result.

$$\text{Garea} = \text{Bytes/Blk} \left\lfloor \frac{\text{Avail Ph1 Storage}}{\text{Bytes/Blk} + \left(\frac{\text{Bytes+Blk}}{\text{Lmin}}\right)(8)} \right\rfloor$$

This total procedure should be followed for each order of merge to produce a maximum and a minimum work area. For utilization of the maximum and minimum, see the introduction in the section <u>Work Area</u>.

Then calculate the number of blocks in a phase-1 sequence.

$$\text{Blks/Seq} = \left\lfloor \frac{\text{Garea}}{\text{Bytes/Blk}} \right\rfloor$$

Finally, determine the number of bytes in a phase-1 sequence.

$$\text{Bytes/Seq} = \text{Blks/Seq} * \text{Bytes/Blk}$$

The required work area can then be calculated:

$$\text{WA} = 4 + 2\left\lceil \frac{\text{NIR(Lmax)}}{\text{Bytes/Seq}} \right\rceil * \left(\frac{\text{Blks/Seq}}{\text{Blks/Trk}} \right)$$

APPENDIX B: CALCULATION OF THE WORK AREA FOR DISK-ADDRESS OUTPUT

To calculate the work area for disk-address output, first, calculate three values (Ph1B1, MaxB2, and MaxB3) using the following formulas:

$$\text{Ph1B1} = \left\lfloor \frac{\text{ST-Sup-Ph1-UA1-BIL-7}}{2(\text{CF} + 8) + 8} \right\rfloor$$

$$\text{Ph1B1} = \left\lfloor \frac{3624}{\text{CF}+8} \right\rfloor$$

The smaller of the two resultant values is the effective Ph1B1 and should be retained. Ph1B1 must not exceed 3624. BIL equals the input block length for fixed-length records or the maximum input block length for blocked or unblocked variable-length records. For an explanation of the other symbols, see <u>Symbol Definition</u>.

$$\text{BL2} = \left\lfloor \frac{\text{St-Sup-Ph2}}{\text{M+1}} \right\rfloor$$

$$\text{MaxBL2} = \left\lfloor \frac{\text{BL2}}{\text{CF+8}} \right\rfloor$$

MaxBL2 is the maximum acceptable phase-2 number of records/block. Retain it for comparison purposes.

$$\text{BL3} = \left\lfloor \frac{\text{ST-Sup-Ph3-UA3-BOL-8-CF}}{\text{M}} \right\rfloor$$

$$\text{MaxBL3} = \left\lfloor \frac{\text{BL3}}{\text{CF+8}} \right\rfloor$$

MaxBL3 is the maximum acceptable phase-3 number of records/block.

Compare MaxBL2 and MaxBL3. The smaller of these two values is the maximum number of records per block (Max Recs/Blk). Compare Max Recs/Blk with Ph1B1. The smaller of these two values is the actual Recs/Blk value.

Calculate the number of Bytes/Blk using the value Recs/Blk just calculated.

Bytes/Blk = Recs/Blk * (CF + 8)

Consult Figure 14 to determine the number of blocks that will fit on a single disk track (Blks/Trk). Next, calculate the number of Bytes/Trk based upon the Blks/Trk just determined.

Bytes/Trk = Blks/Trk * Recs/Blk * (CF +8)

It may be possible to improve the utilization of available disk storage by decreasing the number of Recs/Blk sufficiently to increase the number of Blks/Trk by 1. Refer to Figure 14 to determine the maximum number of Bytes/Blk if the value Blks/Trk is increased by 1. Using the increased Blks/Trk, calculate a decreased number of Recs/Blk (Alt Recs/Blk).

Calculate the decreased Bytes/Blk (Alt Bytes/Blk) with the following equation:

Alt Bytes/Blk = Alt Recs/Blk * (CF + 8)

Next, calculate the total number of bytes (Alt Bytes/Trk) that can be contained on a track if the alternate blocking configuration is used. Use this equation for the calculation:

Alt Bytes/Trk = Alt Blks/Trk * Alt Recs/Blk * (CF+8)

Now, compare the value Bytes/Trk to the value Alt Bytes/Trk. If Alt Bytes/Trk is ≤ the initial Bytes/Trk, use of the alternate blocking configuration <u>will not</u> improve the utilization. However, if Alt Bytes/Trk is > the initial Bytes/Trk, use of the alternate configuration <u>will</u> improve disk utilization.

This process should be repeated for each possible order of merge.

After the values for each order of merge have been determined, the maximum and minimum work areas can be calculated. Using the smallest number of Bytes/Trk that resulted from among the orders of merge considered, determine the maximum number of disk tracks (Amax) that can be used advantageously for the work area.

$$\text{Amax} = 2 * \left\lceil \frac{\text{NIR}*(\text{CF+8})}{\text{Min Bytes/Trk}} \right\rceil + 4$$

Next, select the largest number of Bytes/Trk that resulted from among the Bytes/Trk values and the Alt Bytes/Trk values, and determine the minimum number of disk tracks (Amin) that can be used for the work area.

$$\text{Amin} = 2 * \left\lceil \frac{\text{NIR}*(\text{CF+8})}{\text{Max Bytes/Trk}} \right\rceil + 4$$

Use the formulas in this appendix to calculate the maximum input/output block/record lengths for operations that involve: fixed-length records (those excluded by the assumptions stated in the section Record Lengths and Input/Output Block Lengths); variable-length records; and fixed- or variable-length records with disk-address output.

FIXED-LENGTH RECORDS (SORT)

To determine the maximum input/output record length, compare the results of the following three equations. The smallest result is the maximum acceptable record length. This maximum record length must not exceed 3624 bytes.

Note: For an explanation of the symbols, except OM, see Symbol Definition. OM = 4 if less than 16,384 bytes of main storage are specified. OM = 7 if 16,384 or more bytes of main storage are specified.

1. $A = \left\lfloor \frac{\text{ST-Sup-Ph1-UA1-8}}{2} \right\rfloor$

2. $B = \left\lfloor \frac{\text{ST-Sup-Ph2}}{\text{OM} + 1} \right\rfloor$

3. $C = \left\lfloor \frac{\text{ST-Sup-Ph3-UA3-8}}{\text{OM}+1} \right\rfloor$

To determine the maximum input/output block length, compare the results of the following three equations. The smallest result is the maximum acceptable block length. This block length must not exceed 3624 bytes.

4. $X = \left\lfloor \frac{\text{ST-Sup-Ph1-UA1}}{2 + \frac{8}{L_1}} \right\rfloor$

5. $Y = \left\lfloor \frac{\text{ST-Sup-Ph2}}{\text{OM}+1} \right\rfloor$

6. $Z = \left\lfloor \frac{\text{ST-Sup-Ph3-UA3-8-}L_1+L_3}{\text{OM} + 1} \right\rfloor$

For fixed-length records in variable-length blocks substitute the following formula for equation 4 (X) above.

$$X = \left\lfloor \frac{\text{ST-Sup-Ph1-UA1} + L_1}{3 + \frac{8}{L_1}} \right\rfloor$$

For disk-address output, substitute the following formula for equation 4 (X) above:

$$X = \left\lfloor \frac{\text{ST-Sup-Ph1 - UA1 - 7}}{3 + \frac{8}{(\text{CF} + 8)}} \right\rfloor$$

VARIABLE-LENGTH RECORDS (SORT)

To determine the maximum input/output record length, compare the results of the following three equations. The smallest result is the maximum acceptable record length and must not exceed 3624 bytes.

Note: For an explanation of the symbols, except OM, Lmin, and Lmax, see Symbol Definition. OM = 3 if less than 16,384 bytes of main storage are specified. OM = 6 if 16,384 or more bytes of main storage are specified. Lmin = the smallest record length; Lmax = the largest record length.

1. $A = \left\lfloor \frac{\text{ST-Sup-Ph1-UA1-15}}{3} \right\rfloor$

2. $B = \left\lfloor \frac{\text{ST-Sup-Ph2}}{2(\text{OM})+1} \right\rfloor$

3. $C = \left\lfloor \frac{\text{ST-Sup-Ph3-UA3-8}}{2(\text{OM})+1} \right\rfloor$

To determine the maximum input/output block length, compare the results of the following three equations. The smallest result is the maximum acceptable block length and must not exceed 3624 bytes.

4. $X = \left\lfloor \frac{\text{ST-Sup-Ph1-UA1-7}}{3 + \frac{8}{L\ \text{min}}} \right\rfloor$

5. $Y = \left\lfloor \frac{\text{ST-Sup-Ph2-(Lmax) (OM)}}{\text{OM} + 1} \right\rfloor$

6. $Z = \left\lfloor \frac{\text{ST-Sup-Ph3-UA3-(Lmax) (OM)-8}}{\text{OM} + 1} \right\rfloor$

MERGE-ONLY INPUT/OUTPUT BLOCK SIZES

The merge-only input/output block sizes must conform to this equation:

$$\text{ST} \geq \text{Sup} + \text{Ph4} + \text{UA4} + \text{Output Blocksize} + \text{OM (Input Block Length)} + L_1 - L_3$$

Note: For an explanation of the symbols, except OM, see Symbol Definition.

OM = 1, 2, 3, or 4, depending upon the number of input files to be merged.

The maximum block length must not exceed 3,624 bytes.

This appendix contains sample times derived from estimates produced with the timing program (program number 1401-LM-079) described in the Systems Reference Library publication IBM 1401/1460 Timing Program for IBM Basic Operating System/360 Sort/Merge Program (8K Disk), C24-3377. These sample times are based upon the following assumptions:

1. The main storage capacity specified is the number of bytes of main storage available for sorting.

2. A physical IOCS and supervisor size of 3,500 bytes.

3. Disk input and output.

4. The number of disk units specified refers to the number of units actually available for sorting (2,000 tracks are assumed to be available on each unit). This area includes the input, work, and output areas.

5. The input/output block lengths are normally the maximums for the specified file and system configuration (see Record Lengths and Input/Output Block Lengths). However, to produce estimates for some of the larger file sizes run on 8K or 16K machines, the input and output blocking must be assumed to have a value that will allow the maximum number of records per track in the input and output files. If the specified record length is not a submultiple of the maximum block length, the assumed block length is:

$$\left\lfloor \frac{\text{Max Acceptable BL}}{L} \right\rfloor * L$$

These times are rounded to the nearest minute. If the estimated time is less than one minute, the charted time is shown as one minute.

If no time is charted, the specified file size exceeds the maximum file size for the specified system configuration.

Note: The timing estimates in this appendix do not include the VERIFY option.

Figure 20. Sample Timing Estimate for Model 40

TIMES IN MINUTES FOR IBM SYSTEM/360 MODEL 40 BASIC OPERATING SYSTEM DISK SORT

		CPU CORE STORAGE CAPACITY											
		8192 BYTES			16384 BYTES			32768 BYTES			65536 BYTES		
NO DISK UNITS	NO RECORDS	REC LENGTH IN BYTES			REC LENGTH IN BYTES			REC LENGTH IN BYTES			REC LENGTH IN BYTES		
		50	100	200	50	100	200	50	100	200	50	100	200
	5000				2	4	8	1	2	4	1	2	4
	10000				4	8	20	2	5	9	2	4	7
	20000				8	19		5	9		4	7	
1	30000				15			7			6		
	40000				19			10			7		
	50000												
	60000												
	5000				2	3	5	1	2	3	1	2	3
	10000				3	5	10	2	3	6	2	3	5
	20000				5	10	31	3	6	14	3	5	13
2	30000				8	16		5	9		4	9	
	40000				10	27		6	14		5	13	
	50000				13			8			8		
	60000				15			9			9		
	5000				2	3	5	1	2	3	1	2	3
	10000				3	5	10	2	3	6	2	3	5
	20000				5	10	19	3	6	12	3	5	11
3	30000				8	15	34	5	9	17	4	9	17
	40000				10	19		6	12		5	12	
	50000				13	27		8	15		8	14	
	60000				15	32		9	18		9	17	
	5000				2	3	5	1	2	3	1	2	3
	10000				3	5	10	2	3	6	2	3	5
	20000				5	10	19	3	6	12	3	5	11
4	30000				8	15	30	5	9	17	4	9	17
	40000				10	19	45	6	12	25	5	12	23
	50000				13	24		8	15		8	14	
	60000				15	29		9	18		9	17	
	5000				2	3	5	1	2	3	1	2	3
	10000				3	5	10	2	3	6	2	3	5
	20000				5	10	19	3	6	12	3	5	11
5	30000				8	15	30	5	9	17	4	9	17
	40000				10	19	43	6	12	25	5	12	23
	50000				13	24	58	8	15	31	8	14	28
	60000				15	29		9	18		9	17	
	5000				2	3	5	1	2	3	1	2	3
	10000				3	5	10	2	3	6	2	3	5
	20000				5	10	19	3	6	12	3	5	11
6	30000				8	15	30	5	9	17	4	9	17
	40000				10	19	43	6	12	25	5	12	23
	50000				13	24	54	8	15	31	8	14	28
	60000				15	29	70	9	18	39	9	17	35

Figure 21. Sample Timing Estimate for Model 30

TIMES IN MINUTES FOR IBM SYSTEM/360 MODEL 30 BASIC OPERATING SYSTEM DISK SORT

CPU CORE STORAGE CAPACITY

NO DISK UNITS	NO RECORDS	8192 BYTES			16384 BYTES			32768 BYTES			65536 BYTES		
		REC LENGTH IN BYTES			REC LENGTH IN BYTES			REC LENGTH IN BYTES			REC LENGTH IN BYTES		
		50	100	200	50	100	200	50	100	200	50	100	200
	5000	5	11	24	2	4	8	2	2	5	1	2	4
	10000	11	24	59	4	8	21	2	5	9	2	4	7
	20000	23	71		9	20		5	10		4	8	
1	30000	36			15			8			6		
	40000	72			20			11			8		
	50000												
	60000												
	5000	3	5	9	2	3	5	1	2	3	1	2	3
	10000	5	9	21	3	5	11	2	3	6	2	3	5
	20000	9	21	93	5	11	33	4	6	15	3	6	14
2	30000	14	40		9	17		5	9		5	10	
	40000	21	95		12	28		7	16		6	14	
	50000	26			14			9			9		
	60000	31			17			11			11		
	5000	3	5	9	2	3	5	1	2	3	1	2	3
	10000	5	9	21	3	5	11	2	3	6	2	3	5
	20000	9	21	45	5	11	21	4	6	12	3	6	12
3	30000	14	32	69	9	16	36	5	9	18	5	10	18
	40000	21	42		12	21		7	13		6	13	
	50000	26	53		14	29		9	17		9	16	
	60000	31	79		17	34		11	20		11	19	
	5000	3	5	9	2	3	5	1	2	3	1	2	3
	10000	5	9	21	3	5	11	2	3	6	2	3	5
	20000	9	21	45	5	11	21	4	6	12	3	6	12
4	30000	14	32	68	9	16	32	5	9	18	5	10	18
	40000	21	42	92	12	21	48	7	13	27	6	13	24
	50000	26	53		14	27		9	17		9	16	
	60000	31	72		17	32		11	20		11	19	
	5000	3	5	9	2	3	5	1	2	3	1	2	3
	10000	5	9	21	3	5	11	2	3	6	2	3	5
	20000	9	21	45	5	11	21	4	6	12	3	6	12
5	30000	14	32	68	9	16	32	5	9	18	5	10	18
	40000	21	42	92	12	21	46	7	13	27	6	13	24
	50000	26	53	115	14	27	62	9	17	34	9	16	31
	60000	31	72		17	32		11	20		11	19	
	5000	3	5	9	2	3	5	1	2	3	1	2	3
	10000	5	9	21	3	5	11	2	3	6	2	3	5
	20000	9	21	45	5	11	21	4	6	12	3	6	12
6	30000	14	32	68	9	16	32	5	9	18	5	10	18
	40000	21	42	92	12	21	46	7	13	27	6	13	24
	50000	26	53	115	14	27	58	9	17	34	9	16	31
	60000	31	72	153	17	32	74	11	20	42	11	19	38

INDEX

C24-3321-3

IBM S/360 Printed in U.S.A. C24-3321-3

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

READER'S COMMENT FORM

IBM BOS/360 Sort/Merge Program Specifications C24-3321-3

- Your comments, accompanied by answers to the following questions, help us produce better publications for your use. If your answer to a question is "No" or requires qualification, please explain in the space provided below. All comments will be handled on a non-confidential basis. Copies of this and other IBM publications can be obtained through IBM Branch Offices.

	Yes	No
● Does this publication meet your needs?	☐	☐
● Did you find the material:		
Easy to read and understand?	☐	☐
Organized for convenient use?	☐	☐
Complete?	☐	☐
Well illustrated?	☐	☐
Written for your technical level?	☐	☐

- What is your occupation? ______
- How do you use this publication?

As an introduction to the subject?	☐	As an instructor in a class?	☐
For advanced knowledge of the subject?	☐	As a student in a class?	☐
For information about operating procedures?	☐	As a reference manual?	☐

Other ______

- Please give specific page and line references with your comments when appropriate. If you wish a reply, be sure to include your name and address.

COMMENTS:

- Thank you for your cooperation. No postage necessary if mailed in the U. S. A.

fold fold

FIRST CLASS
PERMIT NO. 387
ROCHESTER, MINN.

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
Systems Development Division
Development Laboratory
Rochester, Minnesota 55901

Attention: Programming Publications, Dept. 425

fold fold

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]