HP RPG/XL Utilities

HP 3000 MPE/iX Computer Systems

Edition 1

Manufacturing Part Number: 30318-90006
E1089

U.S.A. October 1989

Notice

Restricted Rights Legend

Acknowledgments

UNIX is a registered trademark of The Open Group.

Hewlett-Packard Company
3000 Hanover Street
Palo Alto, CA 94304 U.S.A.

HP RPG/XL Utilities-Part 1 XSORT

1 PREFACE

EXTRA FUNCTION SORT FOR RPG (XSORT)

This manual is intended for the use of inexperienced as well as seasoned programmers. To answer the needs of programmers with varied backgrounds, the manual presents information of varying complexity, especially in explaining the XSORT specifications. Most of this manual will be devoted to the five-part specifications. The information will be presented under the following headings:

Overview of XSORT Specifications

shows you which columns of the specifications you must consider when you are preparing a sort. Overviews serve as a quick reminder to those who are familiar with the program

Overview of Column Entries

lists all possible entries for each column in the specifications. These are brief descriptions and explanations of the entries and their purposes. Column Entries

provide detailed discussions of the meanings of all possible entries, their relationships to other entries within the specifications type and other entries in other specification types. They are intended for new users of XSORT.

Using XSORT Commands

gives you the various MPE commands necessary to use XSORT in job stream and interactive session, with different arrangements of input and output files.

XSORT Applications

demonstrate how specifications are adapted to the requirements of particular jobs.

The more extensive the user's experience, the more readily will XSORT's versatility and efficiency become apparent. The manual takes into consideration that some users will be employing the program for the first time. Information and instruction are provided for several levels of experience so that the new user should be able to put XSORT to work after reading the manual.

In using XSORT, you may have programs and systems that interface with the Keyed Sequential Access Method (KSAM/3000), the IMAGE Data Base Management System (IMAGE/3000), and the Data Entry and Forms Management System (V/3000). For information about these systems, please read:

Manual Title
IMAGE/3000 Data Base Management Reference System
KSAM/3000 Reference Manual
VPLUS/3000 Reference Manual

Conventions

NOTATION **DESCRIPTION**

UPPERCASE

Within syntax statements, characters in uppercase must be entered in exactly the order shown, though you can enter them in either uppercase or lowercase. For example:

```
SHOWJOB
```

Valid entries: showjob ShowJob SHOWJOB

Invalid entries: shojwob ShoJob SHOW_JOB

italics

Within syntax statements, a word in italics represents a formal parameter or argument that you must replace with an actual value. In the following example, you must replace *filename* with the name of the file you want to release:

RELEASE *filename*

punctuation

Within syntax statements, punctuation characters (other than brackets, braces, vertical parallel lines, and ellipses) must be entered exactly as shown.

{ }

Within syntax statements, braces enclose required elements. When several elements within braces are stacked, you must select one. In the following example, you must select ON or OFF:

```
       {ON }
SETMSG {OFF}
```

[ ]

Within syntax statements, brackets enclose optional elements. In the following example, brackets around ,TEMP indicate that the parameter and its delimiter are optional:

PURGE {*filename* } [,TEMP]

When several elements with brackets are

stacked, you can select any one of the elements or none. In the following example, you can select *devicename* or *deviceclass* or neither:

SHOWDEV [*devicename*]
        [*deviceclass*]

[...]

Within syntax statements, a horizontal ellipsis enclosed in brackets indicates that you can repeatedly select elements that appear within the immediately preceding pair of brackets or braces. In the following example, you can select *itemname* and its delimiter zero or more times. Each instance of *itemname* must be preceded by a comma:

[*,itemname*][...]

If a punctuation character precedes the ellipsis, you must use that character as a delimiter to separate repeated elements. However, if you select only one element, the delimiter is not required. In the following example, the comma cannot precede the first instance of *itemname*:

[*itemname*][,...]

|...|

Within syntax statements, a horizontal ellipsis enclosed in parallel vertical lines indicates that you can select more than one element that appears within the immediately preceding pair of brackets or braces. However, each element can be selected only one time. In the following example, you must select ,A or ,B or ,A,B or ,B,A :

{,A}
{,B}|...|

If a punctuation character precedes the ellipsis, you must use that character as a delimiter to separate repeated elements. However, if you select only one element, the delimiter is not required. In the following example, you must select A or B or AB or BA. The first element cannot be preceded by a comma:

```
{A}
{B}|,...|
```

...

Within examples, horizontal or vertical ellipses indicate where portions of the example are omitted.

Å

Within syntax statements, the space symbol Å shows a required blank. In the following example, you must separate *modifier* and *variable* with a

blank:

SET[(*modifier*)]Å(*variable*);

underlining

User input is underlined. For

example:

PROMPT?response

In a syntax statement, brackets, braces or ellipses are underlined if you must enter them. For example:

COMMAND

[[*ParameterA*]] = *ParameterB*

Conventions (continued)

NOTATION

DESCRIPTION

shading

Within an example of interactive dialog, shaded characters indicate user input or responses to prompts. In the following example, OMEGA is the user's response to the NEW NAME prompt:

NEW NAME? OMEGA

The symbol   indicates a key on the terminal's keyboard. For example, CTRL indicates the Control key.

CTRL*char*

CTRL*char* indicates a control

character.

For example, CTRLY means you have to simultaneously press the Control key and the Y key on the keyboard.

base prefixes

The prefixes %,#, and $ specify the numerical base of the value that follows:

%*num* specifies an octal number.
#*num* specifies a decimal number.
$*num* specifies a hexadecimal
number.

When no base is specified, decimal is assumed.

2 Introduction

Hewlett-Packard's Extra Function Sort for RPG/XL, hereaft er referred to as XSORT, enables users to rearrange data in their files, drop records from their files, and reformat records, as shown in figure 1-1. XSORT runs on the MPE XL computer system and is primarily used in conjunction with the RPG programming language. Its origin was the unsupported utility SORT3 which was designed to perform most of the functions of IBM's System/3 $DSORT and System/34 #GSORT. Similarities and differences are covered in appendix A.

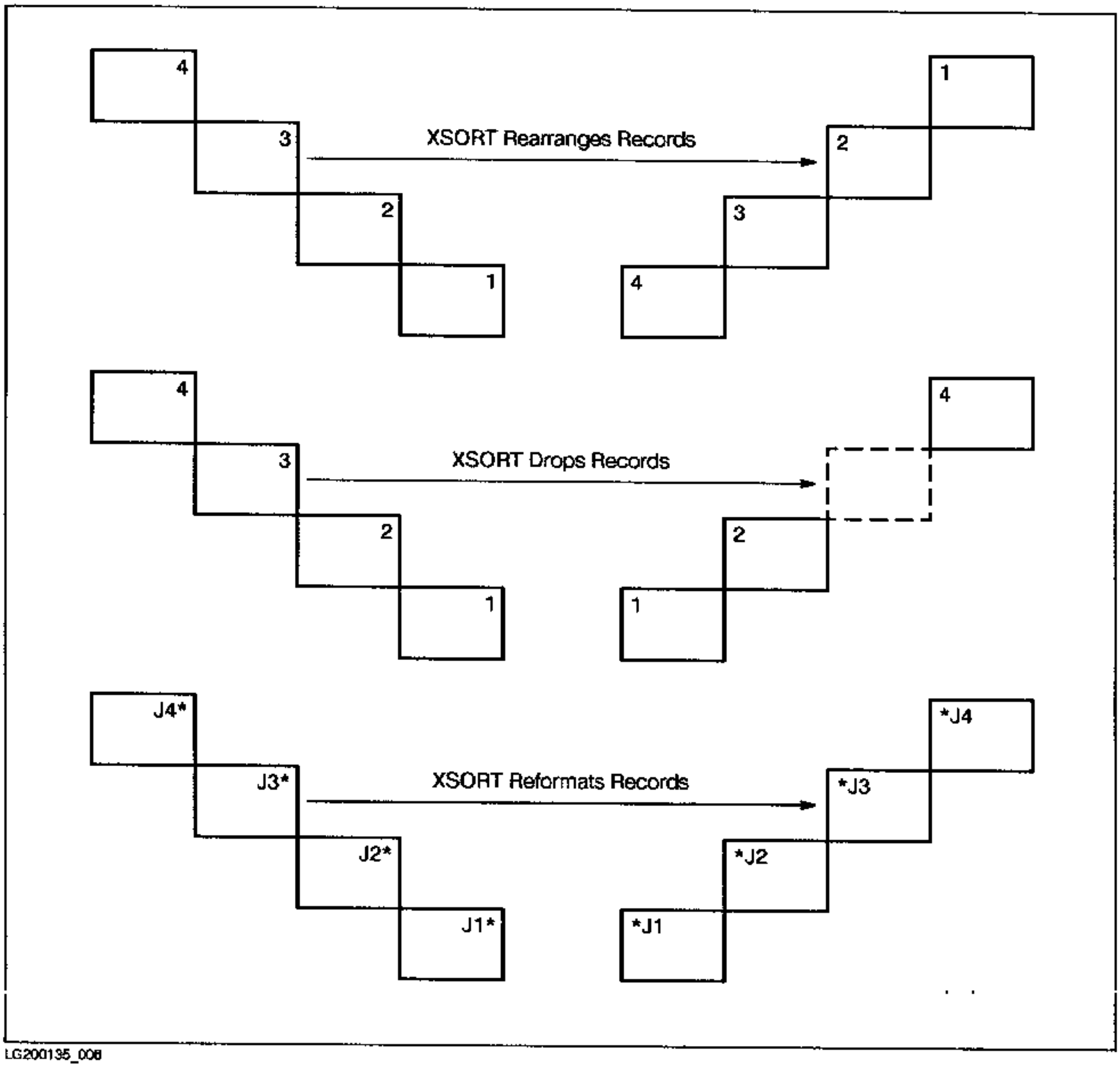


Figure 1-1. Uses of XSORT

Characteristics of XSORT:

In extending and enhancing your sort capabilities, XSORT:

- Uses from one to nine input files.

- Employs multiple logical criteria to select records from MPE (disk and non-disk) and KSAM files, so you can process a subset of your input records.

- Builds a different sort key for each type of record using input record fields and forced control fields. This facilitates the input of files with different data structures.

- Reformats each type of record upon output to give you a choice of record layouts.

- Can perform a count-only pass on the input files. This permits you to conduct a preliminary search of your records or provide statistics on selected subsets of records.

How XSORT Works

XSORT follows a logical sequence in performing a sort. This sequence is described below:

1 XSORT reads a record from the input file.

2 XSORT checks your specifications to make sure the record is one you want to sort. You may not wish to sort all the records in the file you have specified.

3 XSORT builds a work record containing sort control fields and data fields. It also formats the data portion according to your specifications. Work record formatting is important because it controls the format of your output.

4 XSORT submits the work records to the SORT subsystem for sorting.

5 XSORT retrieves records from SORT after sorting is complete.

6 XSORT writes records into the output file, retaining or dropping control fields, according to your specifications

NOTE Control fields are input fields whose characteristics are used as a basis for the comparisons required by the sort. Control fields are discussed in detail in the section on Field Descriptions. (See chapter 6.)

```
-----------------------------------------------------
|                  |                |                |
|  Control field   |      Data      | Control field  |
|       A          |                |       B        |
|                  |                |                |
-----------------------------------------------------
1                10 11            29 30            36
```

Suppose that you wanted to rearrange the input file above because a proposed sort job requires the following changes:

1 Change the contents of positions 30 through 36 placed in positions 1 through 7 of the work record.

2 Change the contents of positions 1 through 10 of the input file placed in positions 8 through 17 of the work file.

3 Change the contents of positions 11 through 29 of the input record placed in positions 18 through

36 of the work record.

The following is an example of how XSORT builds your work record to reflect your specifications:

```
--------------------------------------------------------
|                  |                  |                  |
|  Control field   |  Control field   |       Data       |
|                  |                  |                  |
--------------------------------------------------------
1                 7 8              17 18               36
```

NOTE The control fields must precede the data fields in the work record. They will be retained on output unless you wish to drop them. Reasons for retaining or dropping control fields will be discussed in the Header Specifications section.

Three Types of Sort

Figure 1-2 illustrates XSORT's sort capability--Address-out, Record-out, and Count-only.

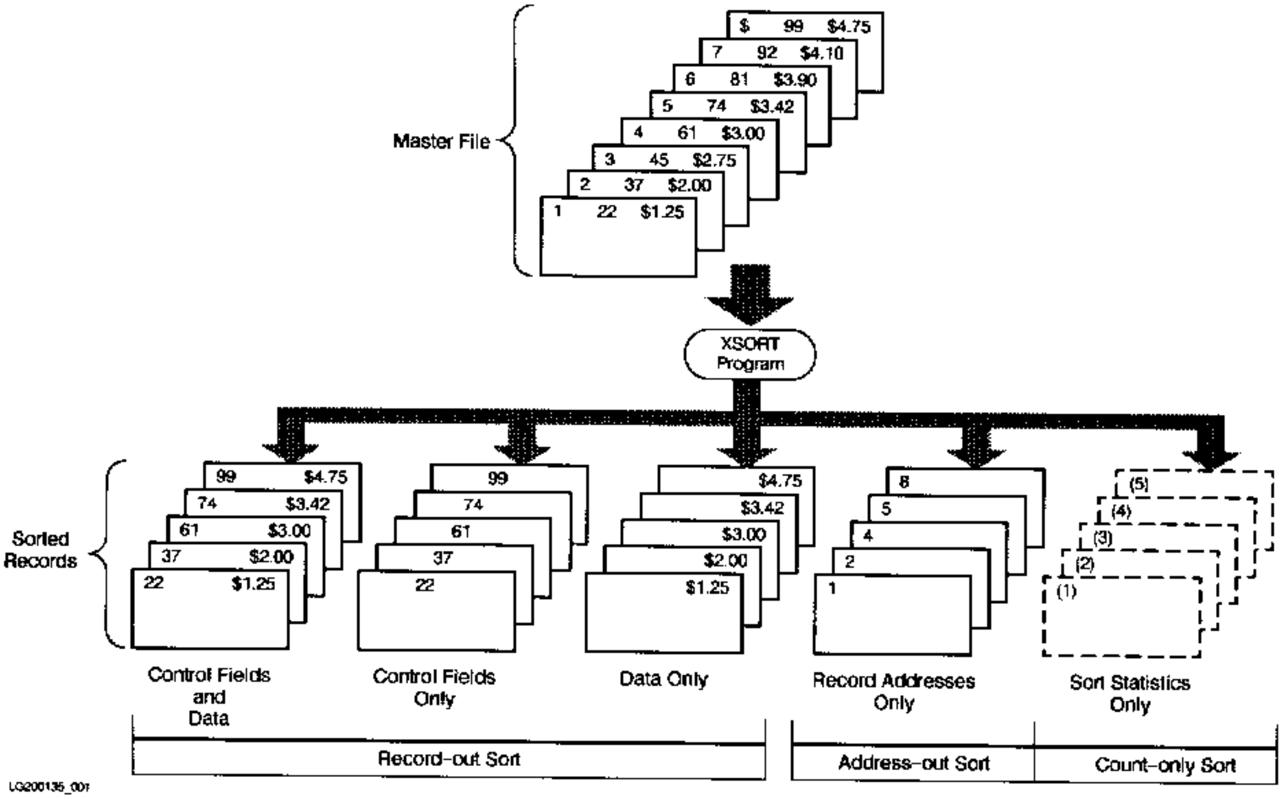


Figure 1-2. Three Types of Sort

Address-Out Sort (SORTA)

The output from an Address-out sort job consists of 4-byte binary relative record numbers of the records in the input file.

An Address-out sort allows you to process a file in sorted order without using up disk or tape space copying the entire file in order to process it.

Record-Out Sort (SORTR)

The output from a Record-out sort is a file of complete records that are immediately available for processing. These sorted records can contain:

- Control fields and data.
- Control fields only.
- Data only.

A Record-out sort offers faster processing than the Address-out as well as enabling you to reformat files and to create new files from reformatted records.

Count-Only Sort (SORTC)

This sort performs a Count-only pass on the input files.

Using a Count-only sort, XSORT can provide you with a statistical analysis of the sort without actually performing the sort or producing an output file.

Running the XSORT Program

The following figure illustrates how XSORT sorts your files.

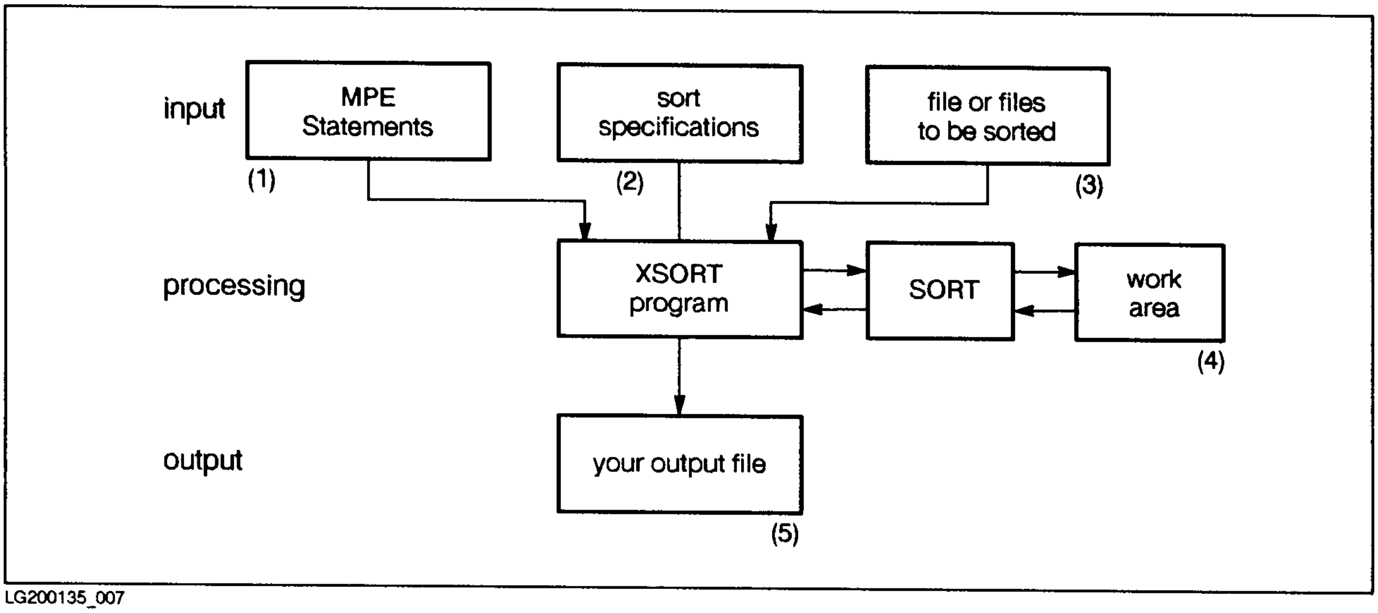


Figure 1-3. How XSORT Sorts Files

1 MPE statements are your job or session control commands to MPE XL.

1 Sort specifications include type of sort, sequence, record information, and instructions formatting output.

2 File or files can be KSAM or sequential MPE (including non-disk files such as magnetic tape).

3 XSORT passes work records to be sorted to SORT which, in turn, sorts them in a work file. Whatever work is required can then be performed without any disturbance to the input files.

4 Output files can contain:

- The relative record numbers of the records in the input file.
- Part or all of the records in the input files.
- A count of the number of records selected for processing from the input files.

Supplying the Sort Specifications

XSORT specifications are normally entered through a job stream. Interactive entry of specs is not recommended because XSORT does not prompt for input. You can use either XSORTEXT or $STDIN to supply sort statements.

XSORT reads your specifications from the file whose formal file designation is XSORTEXT. By default, XSORTEXT is associated with the actual file designator, $STDIN. Specifications are read from XSORTEXT (or $STDIN) and any errors in syntax are detected and communicated to you on $STDLIST.

XSORT specifications must always be entered in the following order.

1 Header.

2 Alternate Collating Sequence.

3 S Option.

4 Record Type.

5 Field Description

HEWLETT PACKARD

XSORT SPECIFICATIONS

Header

Page ______ of ______

Header specifications specify the type of sort job you are running, as well as other general control information.

Line Number	Spec. Type (H)	Type of Sort SORTA, SORTR, or SORTC	Largest sum of control field lengths of any record section (in bytes)	Sequence (A or D)		Seq. (S) Altern	Print	(X) Outp			Comments (any character)
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 68 69 70 71
	H										

← Insert ALTSEQ specifications here (if used).

← Insert S OPTION specification here (if used).

LG200135_031A

Alternate Collating Sequence

ALTSEQ entries specify an alternate collating sequence you wish to use.

A	L	T	S	E	Q	Blank	← Hexadecimal equivalents of paired characters →
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 68 | 69 70 | 71

* * (Asterisks in columns 1 and 2 after last ALTSEQ line)

LG200135_031E

S Option

S Option specifications specify additional control information.

Line Number	Spec. Type (S)	0 or 1	N or Y	Reserved	Maximum number of records to be sorted	Reserved	Comments (Any Character)
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 68 69 70 71
	S						

LG200135_031D

Record Type

Record type specifications specify additional general control information.

Line Number	Spec. Type (I or O)	Continuation (A, O, or *)	C, P, or U	Factor 1 Location from	Factor 1 Location to	Rel. EQ, NE, LT, GT, LE, GE	Factor 2 (Field or Constant) F or C	Factor 2 (Field or Constant) Location from	Factor 2 (Field or Constant) Location to	Factor 2 (Field or Constant) Constant (Any ASCII Character)	Comments (Any Character) Record Name	Comments (Any Character)
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 68 69 70 71

LG200135_031B

Field Description

Field description specifications specify how you want to sort and format the records.

Line Number	Spec. Type (F)	Type (N, O, F, D, or *)	P, U, C, or V	Location from	Location to	Forced Record Charac	Forced Substitute Cha	Forced Continuation	Reserved	Comments (Any Character) Field Name	Comments (Any Character)
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 68 69 70 71
	F										

You do not always need to use all the specification types for every sort; but the order in which the specifications are used doesn't vary. You always fill out Header Specifications first.

The ALTSEQ (Alternate Collating Sequence) and the S (Special) Option specifications can be ignored if they are not to be used in the sort. However, S Option specifications must be supplied when your Input is from non-disk devices or when you are sorting multiple files.

It is not necessary to fill out the Record Type entries if you wish to have all the records in the file sorted and they are all written in the same format.

Number of Records to be Sorted	**Format of Records to be Sorted**	**Order of Specification**
All the records in the file.	All the same.	1. Header** 2. Field descriptions
Some of the records in the file.	All the same.	1. Header** 2. Record Type specs 3. Field descriptions
All or some of the records in the file.	Several different formats (record sections)*	For first record format: 1. Header** 2. Record Type specs 3. Field descriptions For second record forma 4. Record Type specs 5. Field descriptions One set for each additional record format: 6. Record Type specs 7. Field descriptions

* This does not mean that the records in the input file must be grouped in sections by format type. Just the specifications must be grouped (one section of Record Type and Field Description lines for each format type). The input records can be in any order.

** ALTSEQ and S Option, if used, would be entered following the Header Specification.

Forms are provided in appendix D to facilitate your organizing specifications before entering them. The forms include 72 column spaces but entries can be extended to 80 columns. Copies of the forms may be made at your convenience.

Detailed Instructions on how to enter the specifications and the optional entries appear in chapters 2 through 6 of the manual.

MPE Commands and Their Uses

You must supply the necessary commands to your system in order for it to perform the sort job you require. Instructions concerning files, file locations, and the name of the program are typical of the information supplied by MPE commands.

An unsupported program similar to XSORT called SORT3 has been in existence for some time.

A discussion of the commands for SORT3 as compared to XSORT appears in appendix A.

Your MPE Commands Provide:

- Name of the program--XSORT.PUB.SYS
- Name of the files you want to sort. The formal designator for a single-file sort is XSORTIN. For multiple-file sorts, formal file designators are XSORTIN1 through XSORTIN9.
  If XSORT can open the file XSORTIN, it assumes you want to do a single-filesort. If XSORTIN cannot be opened, XSORT assumes you want to do a multiple-filesort, starting with XSORTIN1. (See "Use of the MPE RESET Command" below for a description of how to avoid problems that may occur when running a multiple-file sort after a single-file sort.)
- Name of your output file. The formal designator is XSORTOUT.
- Name of the file containing specifications of the sort. The formal designator is XSORTEXT, which defaults to $STDIN.

Use of the MPE RESET Command:

To ensure that succeeding executions of XSORT use the correct files, it is recommended that you use MPE RESET commands after each use of XSORT to reset all the MPE FILE equations issued for your input, output, and text files.

For example, in a job stream:

```
.....

!FILE XSORTIN=TESTFILE
!FILE XSORTOUT=TESTFILE
!FILE XSORTEXT=TESTSPEC
!RUN XSORT.PUB.SYS
!RESET XSORTIN
!RESET XSORTOUT
!RESET XSORTEXT
....
```

Without the RESETs, the next sort could accidentally use the file TESTFILE for input (if the new sort is a multiple-file sort, naming XSORTIN1 instead of XSORTIN) or TESTSPEC for the sort specifications (if an overriding file equation for XSORTEXT is not issued).

Input Files

XSORT can input sequential MPE files, including files on non-disk devices such as tape. (Please see Section 7 for the commands required.)

KSAM files may be input. Caution should be taken, however, because XSORT will skip deleted records. A deleted record has its first two bytes set to -1 (to hex "FFFF"). For Address-out sorts (SORTA), XSORT reads KSAM files in chronological sequence and determines record numbers accordingly.

XSORT will normally assure that the first record for a KSAM file is record number 0. If you wish to start the Address-out numbering at 1, then you must supply the S Option spec with a "1" in column 7 or build the KSAM file with the Firstrec=1 option. (See Section 4 for further information regarding S Option specification.)

NOTE XSORT will not process IMAGE data base files.

The formal designator for a single-file sort is XSORTIN. For multiple-file sorts, formal file designators are XSORTIN1 through XSORTIN9. The multiple-file designators do not have to be continuous. You can skip file numbers, selecting XSORTIN1, XSORTIN7, and XSORTIN3, for example. However, you must start with XSORTIN1.

When you use more than one input file:

- You cannot do an Address-out (SORTA) sort.

- You cannot do an in place sort, in which output file overlays input file.

- XSORT bases its operations on the characteristics of the first file opened. All the other files are assumed to have the same, or compatible, characteristics (file type and record length).

- You must be sure that all your specifications apply to all the records in all the files you wish to use in a multifile sort.

- You can ensure that you do not accidentally do a single-file sort (instead of a multiple-file sort) by issuing the following MPE command prior to running your XSORT program:

  ```
  RESET XSORTIN
  ```

  (See "Use of the MPE RESET Command.")

If single file input is from $STDIN, the specification records cannot be from $STDIN. They must be from a file referenced by the formal designator XSORTEXT.

Work Space

- XSORT tells SORT how much work space will be required for performing the actual sort. For single-file input from disk, it determines the space needed to sort by the number of records in the input file. You can directly supply this number by specifying the "maximum numbers of records to be sorted", using the S Option (See Section 4). This must be done in the following two cases:

- Single-file non-disk input--including $STDIN.

- Multiple-file input.

If only a small percentage of the records from a single-file sort are being selected and sorted, XSORT will make more efficient use of resources if a reasonably accurate "maximum number of records to be sorted" is specified using the S Option.

Output File

The output file is your sorted file. It can be a sequential MPE file or an already existing KSAM file. The output can be given in one of three formats:

- Address-out.

- Record-out.

- Count-only. (No output file is produced. XSORT only prints sort statistics.)

The formal file designator is XSORTOUT.

If XSORTOUT does not already exist (non-disk files by definition already exist) then XSORT will create it as a job temporary disk file. If XSORTOUT does already exist, then it must be the same file as XSORTIN. In such an instance, the sort is an in place sort. (This requirement can be overridden by coding a "Y" in column 8 of the S Option specs to allow output to an existing file other than XSORTIN.) XSORTOUT can never be the same as any of the XSORTIN1 through XSORTIN9 files in a multiple-file sort.

Data and Timing

Data Considerations

- Maximum data input record size is 8192 bytes.

- The collating sequence may be altered by the ALTSEQ specs. (See appendix C for further information concerning data.)

Timing Factors

The time it takes to run a sort depends on the following factors:

- The number of records to be sorted. The fewer the records specified for sorting, the shorter the sort will take.

- The record size. Larger records take longer to sort.

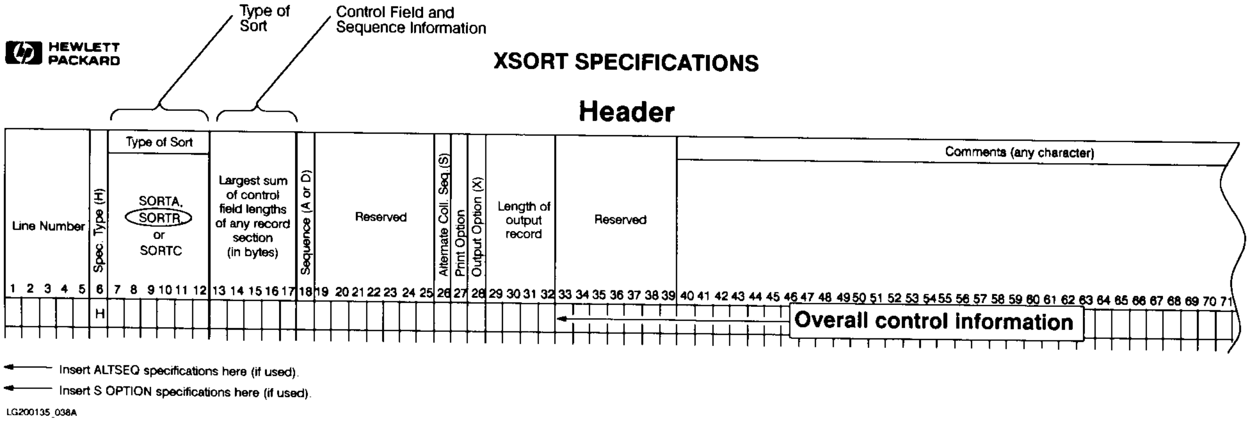
- The number of specifications you enter. The more specifications you have, the longer the sort will take.

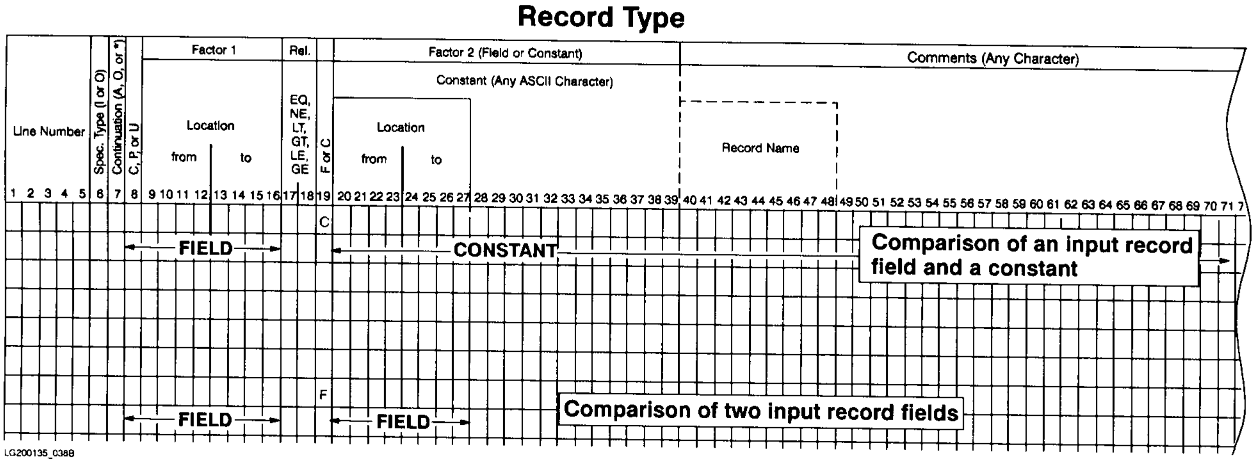
- The alternating sequence. If you use ALTSEQ, your program will take longer to run.

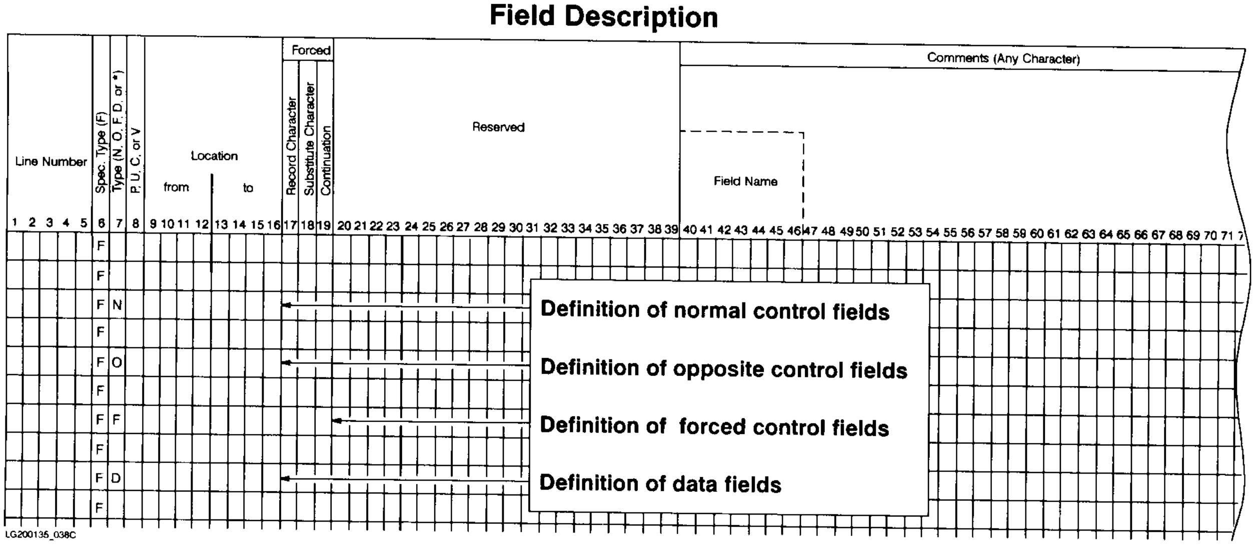
Overview of XSORT Specifications

Specifications for Record-Out Sort (SORTR)

The columns that are shaded must be considered when you are planning a Record-out sort job.

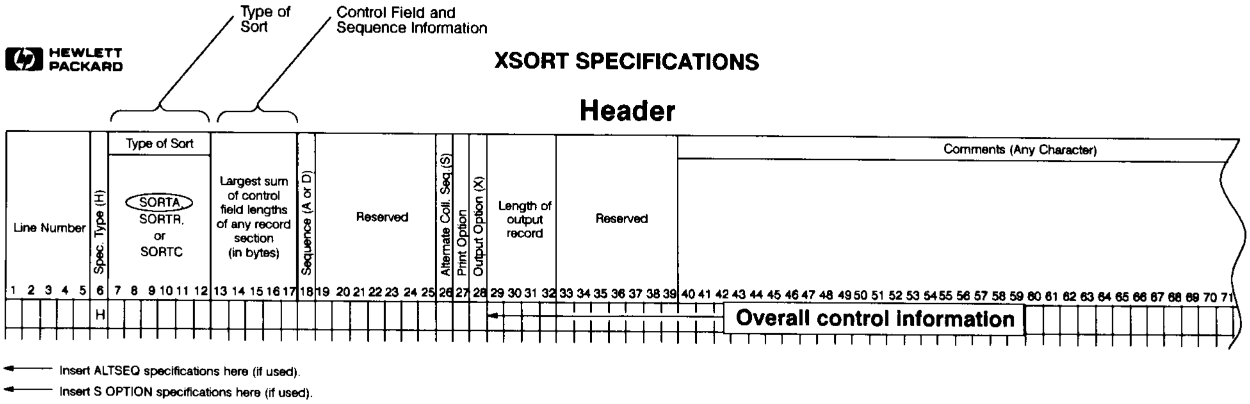






Specifications for Address-Out Sort (SORTA)

The columns that are shaded must be considered when you are planning an Address-out sort job.



HEWLETT PACKARD

XSORT SPECIFICATIONS

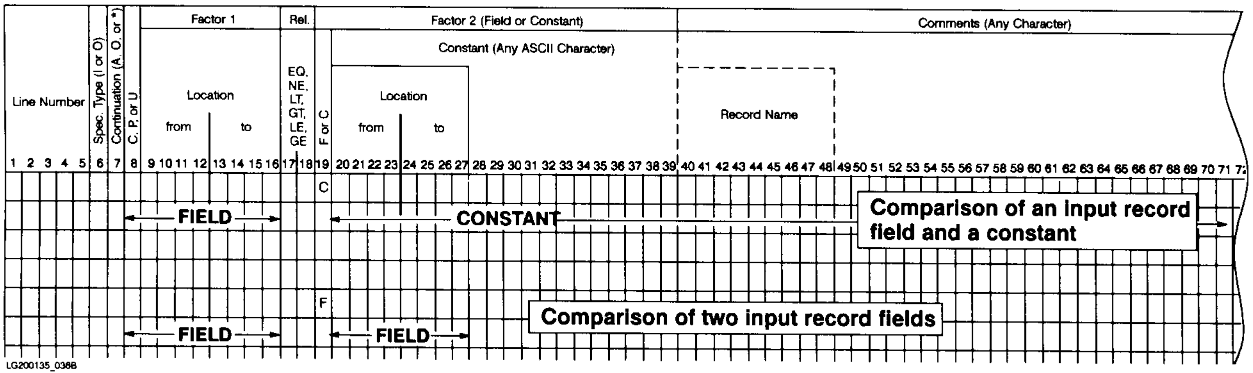
Header

Insert ALTSEQ specifications here (if used).

Insert S OPTION specifications here (if used).

LG200135_036A

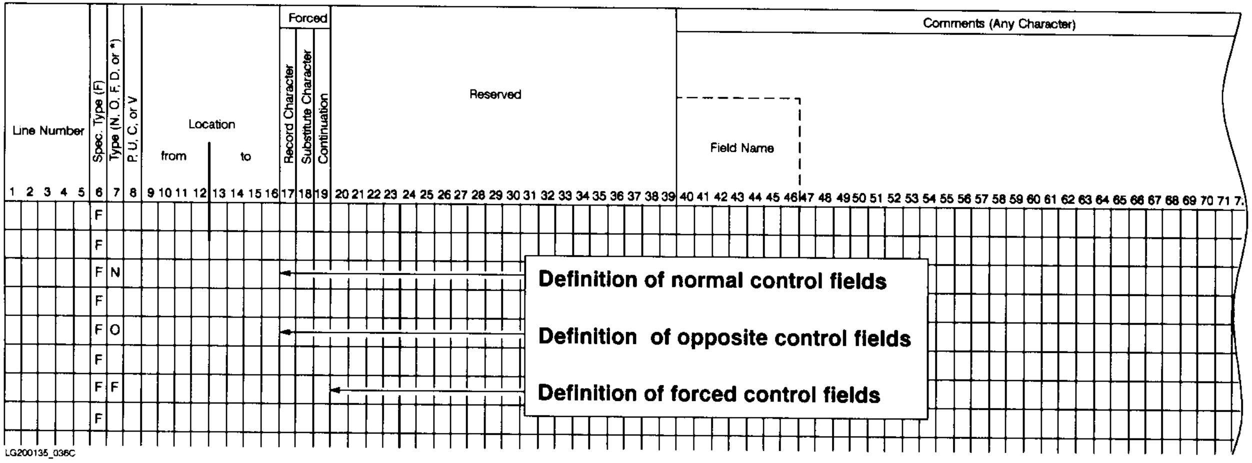
Record Type



Line Number
Spec. Type (I or O)
Continuation (A, O, or *)
C, P, or U
Factor 1
Location
from
to
Rel.
EQ, NE, LT, GT, LE, GE
F or C
Factor 2 (Field or Constant)
Constant (Any ASCII Character)
Location
from
to
Comments (Any Character)
Record Name
C
FIELD
CONSTANT
Comparison of an input record field and a constant
F
FIELD
FIELD
Comparison of two input record fields

LG200135_036B

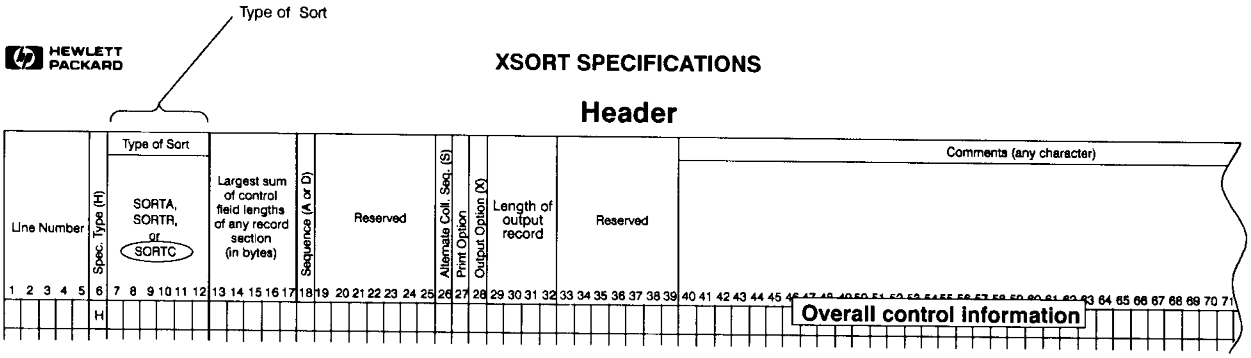
Field Description



LG200135_036C

Specifications for Count-Only Sort (SORTC)

The Columns that are shaded must be considered when you are planning a Count-only sort.

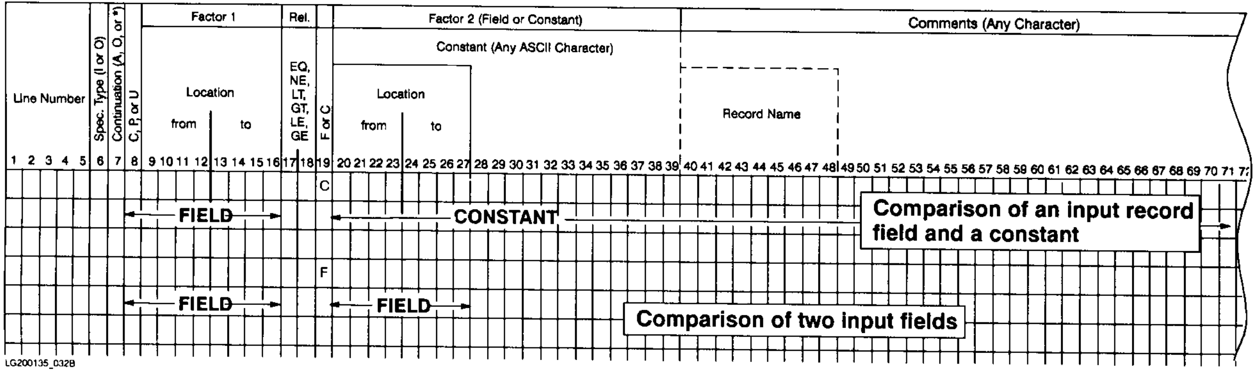


← Insert ALTSEQ specifications here (if used)

← Insert SOPTION specifications here (if used).

LG200135_032A

Record Type



LG200135_032B

3 Chapter 2 Header Specifications

When you enter Header Specifications you are telling XSORT the following:

Introduction

- The type of sort you want to do--Address-out, Record-out, or Count-only.
- The format you have selected for the sorted file.
- The information you wish to have printed to help you check for errors.

Only one Header Specification is needed for each sort job.

Header Specification Column Entries

The following table summarizes the types of column entries.

Table 1: Header Specification Column Entries

Columns	Entries	Explanation
1-5	Any value	Line number.
6*	H	Header line identification.
7-12*	SORTA SORTR SORTC	Address-out sort. Record-out sort Count-only sort.
13-17*	1-256	Largest sum of control field lengths for any Record Section (in bytes).
18**	A D	Records in sorted file to be in ascending order by control fields. Records in sorted file to be in descending order by control fields.
19-25	Blank	Reserved.
26	Blank S	Use standard ASCII collating sequence in compare operations. Use an alternate collating sequence in compare operations. ALTSEQ specs will define the collating sequence desired.

Table 1: Header Specification Column Entries

Columns	Entries	Explanation
27	0 or blank	Print: XSORT specifications Diagnostic messages Statistics Abort messages Display messages
	1	Print: Statistics Abort messages Display messages
	2	Print: Abort messages Display messages
	3	Print only display messages
28***	Blank X	Retain control fields on output records in Record-out sort jobs. Drop control fields from output records in Record-out sort jobs.
29-32**	1-1000	Length of output records in Record-out sorts.
33-39	Blank	Reserved.
40-80	Any Characters	Comments.

* Columns that must be filled in for all sorts.

** Columns that must be filled in for Address-out and Record-out sorts.

*** Columns that must be filled in for Record-out sorts.

Columns 1-5: Line Number

Line number is an optional entry. XSORT does not check your columns 1-5 entries for sequence. Your entry here is intended as a convenience, a record you may use for future reference.

Column 6: Line Type

An entry of H identifies this as the Header line.

Columns 7-12: Type of Sort

In columns 7-12 you tell the program what type of sort you want carried out. SORTA means an Address-out sort. SORTR means Record-out sort. SORTC means a Count-only sort on the input files.

Columns 13-17: Largest Sum

Columns 13 through 17 specify the largest sum of control field lengths of any record section (in bytes).

The entry requires the following calculations:

6 Compute the lengths of all control fields on Field Description specs (when column 7 is N, O, or F--see table below). Sum these totals separately for each record section (see definition of "Sections" on page 5-3).

7 Place the largest of these sums in columns 13-17. The entry cannot exceed 256.

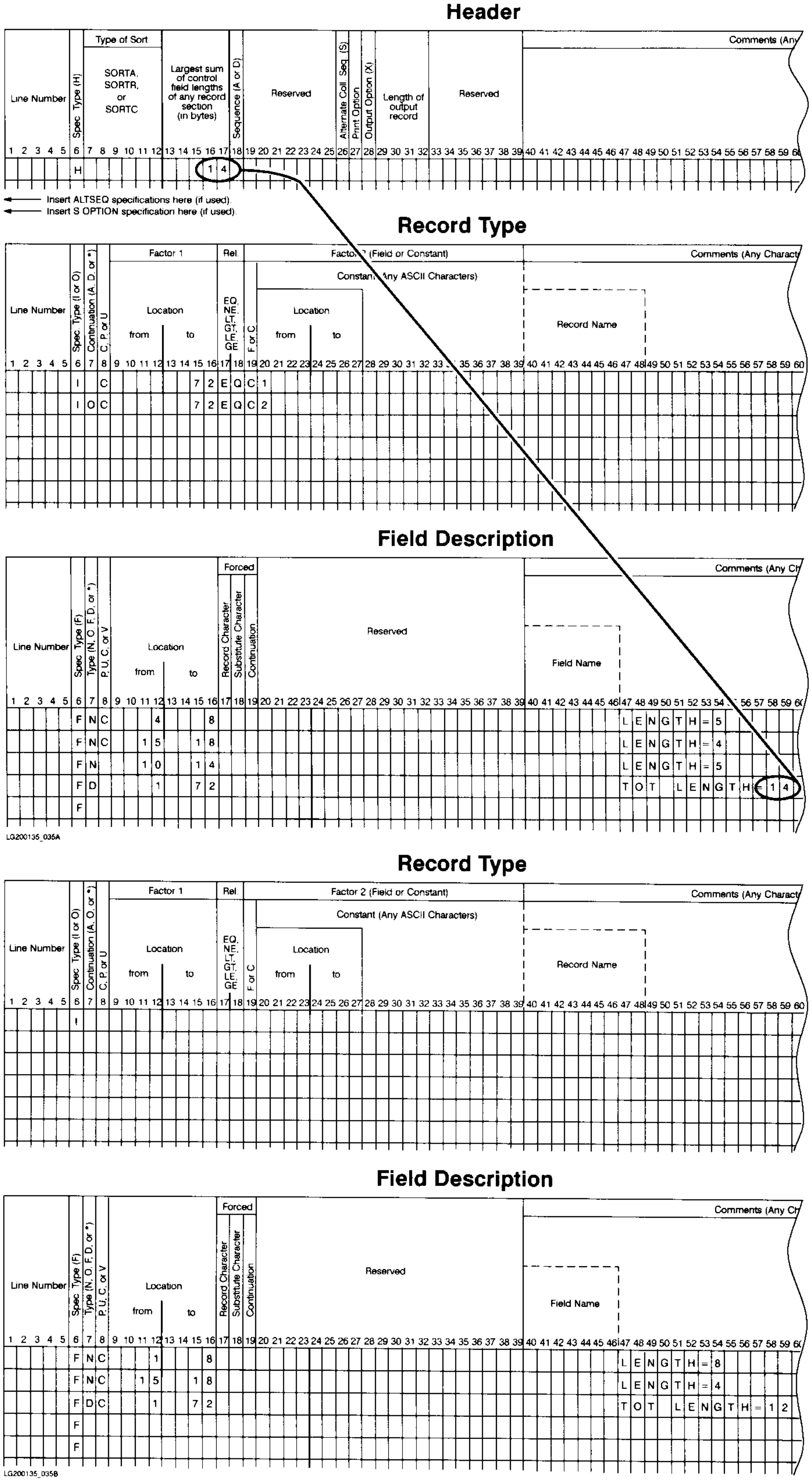
Additional information concerning this entry can be found under column 7 of the Field Description Specifications. Also, see figure 2-1, in which the largest control field sum of two record sections is calculated and the total entered in columns 13-17.

In computing the total lengths of control fields and work records the following guidelines should be noted:

Table 2:

	If Column 7 Entry Is:	**Add This To Total Length:**
Control fields	N or O F	Field length* 1 If column 19 (continuation) is blank. 0 If column 19 (continuation) is non-blank.
Data fields	D	Field length If column 8 is blank. 1 If column 8 contains V.

* Actual length of the field as defined by its input record "location" (columns 9-16).



Header

Line Number (1-5)	Spec. Type (H) (6)	Type of Sort: SORTA, SORTR, or SORTC (7-12)	Largest sum of control field lengths of any record section (in bytes) (13-17)	Sequence (A or D) (18)	Reserved (19-25)	Alternate Coll. Seq. (S) (26)	Print Option (27)	Output Option (X) (28)	Length of output record (29-32)	Reserved (33-39)	Comments (Any (40-60)
	H		14								

◄—— Insert ALTSEQ specifications here (if used).
◄—— Insert S OPTION specification here (if used).

Record Type

Line Number (1-5)	Spec. Type (I or O) (6)	Continuation (A, O, or *) (7)	C, P, or U (8)	Factor 1 Location from (9-12)	Factor 1 Location to (13-16)	Rel. EQ, NE, LT, GT, LE, GE (17-18)	F or C (19)	Factor 2 (Field or Constant) Constant (Any ASCII Characters) / Location from, to (20-39)	Comments (Any Charact / Record Name (40-60)
	I		C		72	EQ	C	1	
	I	O	C		72	EQ	C	2	

Field Description

Line Number (1-5)	Spec. Type (F) (6)	Type (N, O, F, D, or *) (7)	P, U, C, or V (8)	Location from (9-12)	Location to (13-16)	Forced Record Character (17)	Substitute Character (18)	Continuation (19)	Reserved (20-39)	Comments (Any Ch / Field Name (40-60)
	F	N	C	4	8					LENGTH=5
	F	N	C	15	18					LENGTH=4
	F	N		10	14					LENGTH=5
	F	D		1	72					TOT LENGTH=14
	F									

LG200135_035A

Record Type

Line Number (1-5)	Spec. Type (I or O) (6)	Continuation (A, O, or *) (7)	C, P, or U (8)	Factor 1 Location from (9-12)	Factor 1 Location to (13-16)	Rel. EQ, NE, LT, GT, LE, GE (17-18)	F or C (19)	Factor 2 (Field or Constant) Constant (Any ASCII Characters) / Location from, to (20-39)	Comments (Any Charact / Record Name (40-60)
	I								

Field Description

Line Number (1-5)	Spec. Type (F) (6)	Type (N, O, F, D, or *) (7)	P, U, C, or V (8)	Location from (9-12)	Location to (13-16)	Forced Record Character (17)	Substitute Character (18)	Continuation (19)	Reserved (20-39)	Comments (Any Ch / Field Name (40-60)
	F	N	C	1	8					LENGTH=8
	F	N	C	15	18					LENGTH=4
	F	D	C	1	72					TOT LENGTH=12
	F									
	F									

LG200135_035B

Figure 2-1. Calculating Control Field Length

Column 18

Ascending or Descending Sequence

This entry indicates the sequence in which you want the records sorted.

Table 3:

Column 18 Entry	Sequence
A	Ascending sequence by control field
D	Descending sequence by control field

Sequencing will be done according to the ASCII collating sequence unless it is altered using ALTSEQ specifications.

Column 26: Collating Sequence

This entry determines the collating sequence you want XSORT to use in its comparing operations. Characters will be compared to ascertain if one is greater than, equal to, or less than another.

If you leave the column blank, XSORT will use the standard ASCII collating sequence.

An S in column 26 means you want to change the standard collating sequence. To do this you must include a set of ALTSEQ specifications after the Header specification. (See Section 3.)

If you specify an alternate collating sequence, you cannot use packed data for record selection factors or sort control fields. In other words, you cannot enter a P in column 8 of your Record Type specifications or in column 8 of your Field Description specs used to define control fields.

Column 27: Print Option

XSORT can print:

- XSORT specifications.
- Diagnostic messages--for syntax errors in the specifications.
- Statistics of the sort--record counts and timing data.
- Abort messages--a "sortstone" identifying the primary cause of the failure.
- Display messages--MPE abort messages and unnumbered diagnostic messages.

Column 27 entries indicate which of the print options you want to have printed during the job:

Table 4:

Column 27 Entry	Information Printed
0 or blank	XSORT specifications, Diagnostic messages, Statistics, Abort messages, Display messages
1	Statistics, Abort messages, Display messages
2	Abort messages, Display messages
3	Display messages

Column 28: Output Option for Record-Out Sort

Column 28 only applies to Record-out sorts (SORTR). It indicates whether or not you want to drop control fields from output records after they are sorted.

An X means drop them. Leaving Column 28 blank means keeping the control fields.

Reasons for Dropping Control Fields

If you are using an alternate collating sequence or opposite control fields you would probably drop the control fields. In each case the control information is so altered during the sorting process that it is of no use.

You can, however, output those fields in a meaningful form by describing them twice; once as control fields and once as data fields. Since data fields are not involved in the sorting process, they are not changed by the program.

Column 29-32: Output Record Length for Record-Out Sorts

Columns 29-32 apply to Record-out (SORTR) jobs only. The entry in these columns tells the program the length of records in the final sorted file.

When control fields are not being dropped, the length must include both control and data field lengths. Include only the data field lengths when the control fields are being dropped. (Note that the sum of your control field and data field lengths should not exceed the maximum work record length limit of 1000 bytes.)

Calculating Output Record Length Fields

When you are dropping the control fields, total the lengths of the data fields included in the job for each record section (see table on page 2-3). Select the largest total and enter the number in columns 29-32.

When you are not dropping the control fields, total the lengths of the data fields for each record section. Select the largest total and sum this with the entry in columns 13-17 of the Header before entering into columns 29-32.

Column 33-39

Reserved for future use.

Column 40-80: Job Description Comments

Columns 40-80 are reserved for your comments. You can use any characters you want to in these columns. If your column 27 entry calls for the printing of specification lines, your comments here will be printed. Your comments have no effect on your program.

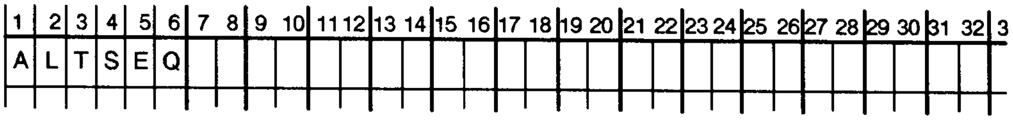
4 Chapter 3 Alternate Collating Sequence Specifications

Normally, XSORT uses the standard ASCII collating sequence included in appendix C. If you want to change the standard collating sequence, you must code an S in column 26 of the Header Specification and code a set of ALTSEQ specifications. You can code whatever number of ALTSEQ lines you need. Each of your statements must begin with ALTSEQ and contain no more than 80 column entries per line.

Users accustomed to a 96 column line will have to convert their programs to the 80 column format. However, it is not necessary to fill in all 80 columns of one ALTSEQ line before continuing on to the next. As long as you enter the ALTSEQ in the first six columns, and submit your characters in pairs as described below, you can enter ALTSEQ characters without filling out all 80 column positions of a line. (The specification forms show only 72 columns but 80 columns per line are allowed.) If you specify an alternate collating sequence, you cannot use packed data for record selection factors or sort control fields. Do not enter a P in column 8 of your Record Type specifications or n column 8 of your Field Description specs used to define control fields.

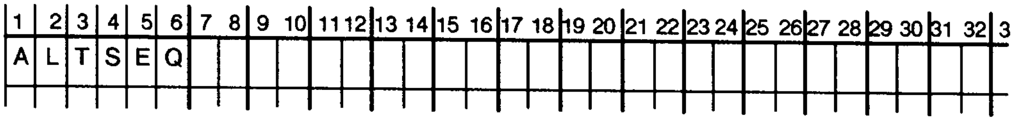
Rules for Coding ALTSEQ

1 Code ALTSEQ in the first six positions. This tells XSORT that this is an ALTSEQ specification.



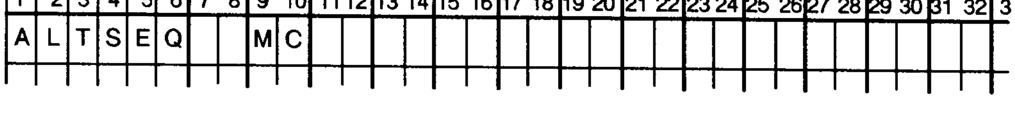
LG200135_008

2 Leave columns 7 and 8 blank.



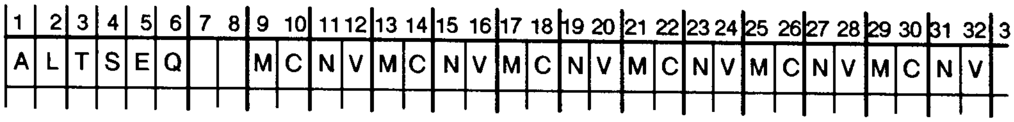
LG200135_009

3 Enter the hexadecimal equivalent of the character you are taking out of normal sequence in columns 9 and 10. (A standard collating chart showing the hex equivalents of all ASCII characters appears in Appendix C.)



LG200135_010

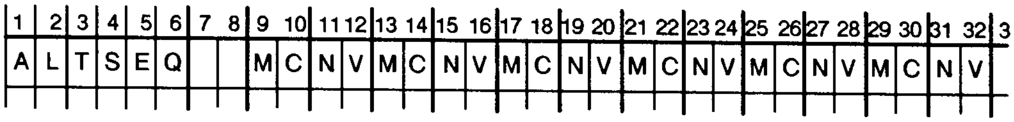
4 Enter in 11 and 12 the hexadecimal equivalent of the value (position) the character specified in columns 9 and 10 is to have in the new collating sequence.



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	3
A	L	T	S	E	Q			M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	

LG200135_012

5 Enter as many such pairs as the number of characters you are taking out of normal sequence.

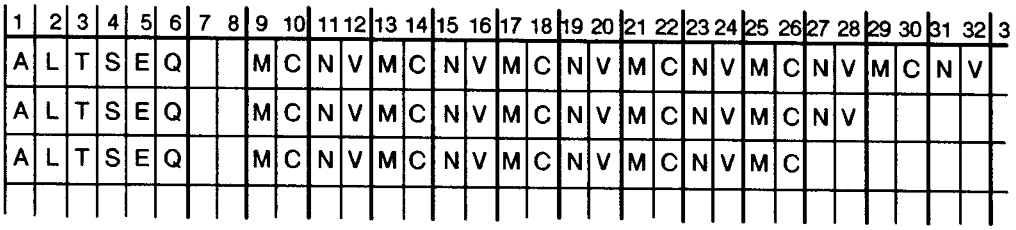


1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	3
A	L	T	S	E	Q			M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	

LG200135_012

6 Leave no spaces between sets of hex numbers.

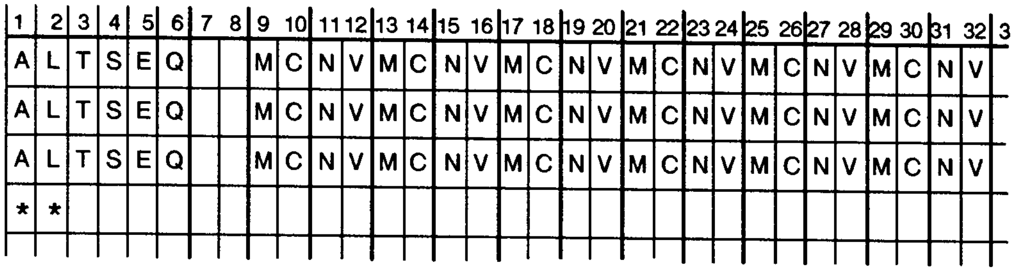
7 When you reach the end of a statement, continue on the next specification line, following the rules above (1-6). Entries can extend to column 80, although it is not necessary to fill an entire line before continuing to a new line.



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	3
A	L	T	S	E	Q			M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	
A	L	T	S	E	Q			M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V					
A	L	T	S	E	Q			M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C							

LG200135_013

8 Enter a double asterisk (**) in columns 1 and 2 of the specification line. This indicates that you are finished with your ALTSEQ statements.



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	3
A	L	T	S	E	Q			M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	
A	L	T	S	E	Q			M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	
A	L	T	S	E	Q			M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	M	C	N	V	
*	*																															

LG200135_014

*MC=moved character, NV=new value

Specs Order Using ALTSEQ

When you use an alternate collating sequence your input must follow an established order:

1 MPE commands.

2 XSORT Specifications.
a Header.
b ALTSEQ entries, including the double asterisks (**).
c S Option, if used.
d Record Type or Field Descriptions as they are required.

Programming with ALTSEQ

Placing an alternate character in the sequence position which is normally assigned to another means that both occupy the same position and are considered equal. (See Sample ALTSEQ Statements section in this chapter.)

If you do not want to have the characters regarded as equal, the one which normally occupies that position must be moved.

Effect of ALTSEQ Statements on Other Coding

ALTSEQ statements can change:

1. Factor 1 and factor 2 (See Record Type Specifications, Section 5).
2. Normal and opposite control fields.
3. Characters in control fields before they are replaced by forced fields.

ALTSEQ statements do not change data fields in records or forced control field characters.

Sample ALTSEQ Statements

1. If you wished to make blanks and zeros equal in your collating sequence you would enter the following:

   ```
   ALTSEQ 2030
   ```

   Blank (20) is moved to the position occupied by zero (30). Blanks and zeros are, therefore, considered equal.

2. If you wished to change the sequence of several characters you should do the following:

   ```
   ALTSEQ 2A4545464647
   ```

   2A45 means * (2A) will occupy the position normally occupied by E (45).
   4647 means F (46) will occupy the position normally assigned to G (47).
   The collating sequence, from character ")" through "H", would be changed:

   ```
    from:   )*+.....ABCDEFGH
                        F
   to:     ) +.....ABCD*E H
                       G
   ```

In this example, F and G would occupy the same position.

5 Chapter 4 Special Option Specification

The Special Option Specification presents you with additional run time options. Special Option, referred to as S Option, can be thought of as an extension of your Header Specification. It offers global controls including:

- Address-out file first record number.
- In place sort override.
- Maximum number of records to be sorted.

The S Option is entered following your Header Specification and ALTSEQ specifications (if used). It can be omitted if you have no need to exercise any of the options, in which case all options default to blank entries. However, it must be used when your input is non-disk or multiple-file (to specify the maximum number of records to be sorted).

Column Entries

The following table gives an overview of column entries and their positions.

COLUMN	ENTRIES	EXPLANATION
1-5	Any Valu	Line number.
6	S	S Option will be exercised
7	0 or blank 1	Numbering for Address-out file starts at 0* Numbering for Address-out file starts at 1.
8	N or blank Y	XSORTOUT must be new or be the same file as XSORTIN XSORTOUT may exist and be a separate file from XSORTIN.
10-16	1-9999999 or blank	Maximum number of records to be sorted.
17-39	blank	Reserved.
40-80	Any Characters	Comments.

* If this column is blank and you are sorting a KSAM file built with Firstrec=1, then record numbering in the Address-out file automatically starts with 1.

Columns 1-5: Line Number

Line number is an optional entry. XSORT does not check your columns 1-5 entries for sequence. Your entry here is intended as convenience, a record you may use for future reference.

Column 6: Line Type

An entry of S identifies this as the S option specification.

Column 7: Address-Out File First Record Number

This entry states the first record number of an Address-out file. It is used primarily for sorts converted from other systems, such as IBM, that start record numbering at 1 instead of 0 (as HP does).

- If you enter blank, the record numbering starts at 0 unless you are sorting a KSAM file built with Firstrec=1, in which case record numbering starts as 1.
- If you enter "0", the record numbering starts at 0. This is true even if you are sorting a KSAM file built with Firstrec=1.
- If you enter "1", the record numbering starts at 1.

Column 8: In Place Sort Override

This entry tells XSORT whether or not you want to override the in place sort rule. Normally, XSORT protects you from accidently, overlaying an existing file with the sort output, XSORTOUT. To do this, it enforces the rule that XSORTOUT, if it already exists, must be the same file as XSORTIN, and this must be an in place sort.

Your column entries determine whether or not you wish the rule to apply in the sort you are running.

Column 8 Entry	XSORTOUT File Already Exists	XSORTOUT File Does Not Exist
N or blank	XSORTOUT must be the same file as XSORTIN. (This is an in place sort.) temporary file	New file will be created for XSORTOUT. It will be a job/session
Y	XSORTOUT may be a different file than XSORTIN. It will be overlaid by sort output records.	New file will be created for XSORTOUT. It will be a job/session

Columns 10-16: Maximum Number of Records to be Sorted

Here you enter the maximum number of records to be sorted. Your entry can range from 1 to 9999999. If omitted, this value defaults to the number of records in the input file. An entry here is required if this is a multiple-file sort or if input is from a non-disk device. It is recommended that you specify this entry in cases where you expect only a small subset of records to be selected for processing from a large input file. This reduces the size of the SORT work file and results in more efficient use of disk space.

Columns 17-39

Columns 40-80

Comments

Columns 40-80 are reserved for your comments. You can use any characters you want to in these columns. If your column 27 entry calls for the printing of specification lines, your comments here will be printed. Your comments have no effect on your program.

6 Chapter 5 Record Type Specifications

Introduction

Record Type Specifications tell XSORT which records in a file you wish to have sorted. If you want all the records in a file to be sorted, and they have all been prepared in the same format, you do not have to fill out this section of your XSORT specifications. Often, however, you use these entries to select records for the sort based on comparisons. Your Record Type Specifications will instruct XSORT to compare data in one field (factor 1) with a constant value or data in another field on the same record (factor 2) to decide whether to include (select) or omit (reject) the record for processing.

Record Types

Data records from the input files may be identified by "type" according to characteristics defined in XSORT Record Type specifications. All records in any Record Type have one or more unique characteristics in common--for example, a 3 in position 5 AND not an X in position 80. Multiple characteristics used to identify one Record Type are tied together using the AND operation ("A" in column 7 of Record Type specifications).

Sections

A "section" is a subset of the input records which is defined by naming one or more Record Types to be included or omitted from it. All records included in a section will be sorted and formatted in the same manner, according to Field Description specifications. The term "section" also refers to the XSORT specifications used to define the section. These consist of Record Type lines which identify the Record Types of the section, followed immediately by Field Descriptions, showing how the section is to be sorted and formatted on output. These sections of specs (one or more) must all follow the Header, ALTSEQ, and S Option specifications.

Multiple Record Types named for one section are tied together using the OR operation ("O" in column 7 of Record Type specifications) whenever several include sets or several omit sets are defined consecutively. (See discussion of sets below.)

The specification sections are used in order of their appearance to identify and categorize each input record. The first section to "accept" an input record for processing is the one used to determine how it will be sorted and formatted. If a section "rejects" the input record, XSORT moves on to the next section to attempt identification. If no section "accepts" the record, it will not be included in any sort processing.

Include-all

is a section of specifications consisting of a single Record Type line containing "I" in column 6 and blanks in columns 7 through 39, followed by Field Description lines. This says that all remaining input records are to be included in this section unconditionally. If used, it must be the last line of the last section defined for the sort job. Include-all is generally used as a "catch-all" mechanism to identify any records not already accepted for processing by any of the preceding specification sections.

Implied Include-all

is a section of specifications consisting only of Field Descriptions. When used, it must be the only specification section defined for the job. It says that all input records are to be included in the sort unconditionally.

Sets

The unique identification of a Record Type is made using sets of "include" and "omit" Record Type Specifications. These sets may be used singly or intermixed to include or omit data records in the current section according to their characteristics. (Multiple characteristics within a set must be tied together using the AND operation.)

An Include set

defines the characteristics of a Record Type that is to be included (accepted) in the current section. (Consecutive include sets must be tied together using the OR operation on the first line of the set.)

An Omit set

defines the characteristics of a Record Type that is to be omitted (rejected) from the current section. (Consecutive omit sets must be tied together using the OR operation on the first line of the set.) Within a section, include/omit sets are used in order of their appearance to evaluate the current input record. If the record matches the characteristics defined for a set (Record Type), it is immediately "accepted" or "rejected" for inclusion in the section, and so no more include/omit sets will be evaluated. If the record does not match the set's characteristics, XSORT moves on to the next include/omit set for evaluation.

Additional information concerning record types, sets and sections and an example of how they are used appear in Section 8. This information is primarily useful for advanced sorts. Typically, users are concerned with simple identification of different Record Types for inclusion or omission.

Column Entries

The following table is an overview of column entries for record type specifications.

COLUMNS	ENTRIES	EXPLANATION
1-5	Any value	Line number.
6*	I O	Include line. Omit line
7	A O Blank *	AND line (these specifications continue the definition of the record which was described on the previous line). OR line (these specifications define a record different than the one described on the previous line) First line of a set of I or O Record Type lines. Comment lines.
8	C P U	Character equals 1 byte (8 bits) of data. Packed decimal data. Unpacked decimal data.
9-12	1-1000	The input record position in which the factor 1 field begins (blank if the field is only one position long).
13-16*	1-1000	The input record position in which the factor 1 field ends.

COLUMNS	ENTRIES	EXPLANATION
17-18*	EQ NE LT GT LE GE	Factor 1 must equal factor 2. Factor 1 must not equal factor 2 Factor 1 must be less than factor 2. Factor 1 must be greater than factor 2. Factor 1 must be less than or equal to factor 2. Factor 1 must be greater than or equal to factor 2.
19*	C F	Factor 2 is a constant. Factor 2 is another field in the same input record.
20-23	1-1000	The input record position in which the factor 2 field begins (blank if the field is only one position long).
24-27	1-1000	The input record position in which the factor 2 field ends.
20-80	Any character	The factor 2 constant.
40-80	Any character	Comments.

* Columns that must be filled in.

Columns 1-5: Line Number

Line number is an optional entry. XSORT does not check your columns 1-5 entries for sequence. Your entry here is intended as a convenience, a record you may use for future reference.

Column 6: Line Type

Your column 6 entry identifies the type of Record Type line you intend to use in your sort. An I in the column stands for an include or include-all line. An O stands for an omit line. Include lines identify the records you want the program to sort by describing particular record fields.

If your sort requires the coding of Record Type lines, you must use at least one include or an include-all line to describe the records you want sorted. Even when you use omit lines, you will have to follow them with an include line. Records not described by your include lines will not be sorted.

Omit lines identify records you do not want the program to sort. Omit lines are not required but they can be helpful when you have many types of records you want to use and just a few you want omitted. Omit lines must be followed by at least one include line or an include-all line. This ensures that the program will check all records not described by an omit line.

Column 7: Continuation or Comments

Column 7 indicates the start or continuation of a Record Type include or omit set definition.

Column 7 Entry	Explanation
Blank	This line contains the first characteristic of a new Record Type that has a different spec type (column 6) than the previous line.
A	This line contains additional characteristics for the same Record Type as that on the previous line. The A stands for AND.

Column 7 Entry	Explanation
O	This line contains the first characteristic of a new Record Type that has the same spec type (column 6) as the previous line. The O stands for OR.
*	The asterisk indicates a comment line. Comment lines are included as an aid in remembering why you were proceeding as you were at various points in the program. The comments will be printed if you leave column 27 of the Header specification blank or enter a zero.

The current Record Type can be further described using additional AND lines, or a new Record Type can be started after this one using an OR line "BLANK" line (in column 7), provided all Record Types are in the same section.

Columns 6 and 7: Relating Entries

Because your column 6 and 7 entries often combine to complete include-omit specifications, the information below is intended to show how to use combinations for the two entries.

Include Sets

Type	Colum n 6	Colum n 7	Explanation
Include AND Lines	I I	blank A	First characteristic of a new Record Type. (Spec type different than previous line.) Additional characteristics for the same Record Type as the previous line.
Include OR Lines	I I	.... O	One or more lines defining the characteristics of one Record Type. First characteristic of a different Record Type than the previous line.
Include AND and OR Lines	I I I	.... O A	One or more lines defining the characteristics of one Record Type. First characteristic of a different Record Type than the previous line. Additional characteristics for the same Record Type as the previous line.
Include-All Line	I		Sort all of the records that have not been described by any preceding include or omit sets. Only one include-all set may be defined per sort job.

An include set can be preceded by the Header (or ALTSEQ or S Option, if used), Field Descriptions for the previous section, or an omit set. An include set can be followed by another include set, an omit set, or Field Descriptions for the current section.

Omit Sets

Type	Colum n 6	Colum n 7	Explanation
Omit AND Lines	O O	blank A	First characteristic of a new Record Type. (Spec type different than previous line.) Additional characteristics for the same Record Type as the previous line.
Omit OR Lines	O O	.... O	One or more lines defining the characteristics of one Record Type. First characteristic of a different Record Type than the previous line.

Type	Column 6	Column 7	Explanation
Omit AND and OR Lines	O	....	One or more lines defining the characteristics of one Record Type.
	O	O	First characteristic of a different Record Type than the previous line.
	O	A	Additional characteristics for the same Record Type as the previous line.

An Omit set can be preceded by the Header (or ALTSEQ or S Option, if used), Field Descriptions for the previous section, or an include set. An Omit set can never be followed by Field Description lines. It must be followed by an include set so that the program will know which of the remaining (non-omitted) records you wish to have sorted.

Mixing Include and Omit Sets

You may find occasions when mixing include and omit sets would be helpful. The program readily accepts the mix, but caution should be exercised since XSORT will accept and respond to commands in the order they are coded. You may, for instance, wish to omit records with a 1 in position 3 while including those with a 1 in positions 3 and 5. You would have to specify your include set first. If you began with your omit set, you would eliminate all the records you wished to sort.

Column 8 Data Type

The Column 8 entry tells XSORT how to interpret data during the compare operations. To do this, the program must first be told the format of the data that is to be compared.

XSORT accepts data in three formats:

Character (C) A character is an 8-bit pattern (byte) which represents a single alphabetic, numeric, special, or control symbol based on the American Standard Code for Information Interchange (ASCII).

Packed (P) Packed numerical data is compacted by storing two digits in each byte. Each digit is stored as a 4-bit binary number with the sign represented by a 4-bit sign code placed in the rightmost, low order position in the field.

Unpacked (U) Unpacked numerical data is represented by a subset of the full ASCII character set, consisting only of the numeric characters (0-9) and the special sign characters for the rightmost digit of numeric fields--positive ({,A-I) and negative (} ,J-R).

Importance of the Column 8 Entry

Through your Record Type Specifications you tell XSORT which records you want sorted. You do so by instructing it to compare data in one field--factor 1--against a constant or data in another field on the same record--factor 2. The result of the comparisons determines whether or not the record will be sorted. The sort program sees your data as nothing more than a series of electronic bits. It must be told how to interpret the data:

- Whether the data is numeric or character.
- If it is numeric, whether it is packed or unpacked.

Column 8 Entry	Comparison Type	Maximum Field Length*
C	Character.	256 characters
P	Numeric data is packed.	14 bytes (27 packed digits and sign)**

Column 8 Entry	Comparison Type	Maximum Field Length*
U	Numeric data is unpacked.	28 characters (unpacked digits)

* Maximum field length applies to both factor 1 and factor 2 data.

** If you specify an alternate collating sequence, you cannot use packed data for record selection factors or sort control fields. Do not enter a P in column 8 of your include or omit Record Type Specifications or in column 8 of your Field Description specs used to define control fields.

NOTE See Appendix C for charts which will be helpful in data formatting.

Columns 9-16: Factor 1 Field

Factor 1 field identifies your record. For example, all your inventory records may contain a 2 in column 2. That would make column 2 a factor 1 field. By comparing factor 1 fields against constants or other fields in the same record, XSORT identifies records you want sorted. Columns 9-16 identify the locations of factor 1 fields in the records. You may at times have more than one factor 1 field. When there is more than one, you must do the following:

- Describe each field in a separate Record Type line.
- Put an A (for AND) in column 7 for all but the first field. This will identify those following as applying to the same Record Type.

Length of the Factor 1 Field

Each factor 1 field can contain from 1 to 256 characters. A factor 1 field cannot be longer than the records you are working with, however. The length of your input records determines how long your factor 1 field can be. Your column 8 entry also controls the maximum length of the factor 1 field.

Column 8 Entry	Maximum Factor 1 Field Length
C	256 characters *
P	14 characters **
U	28 characters

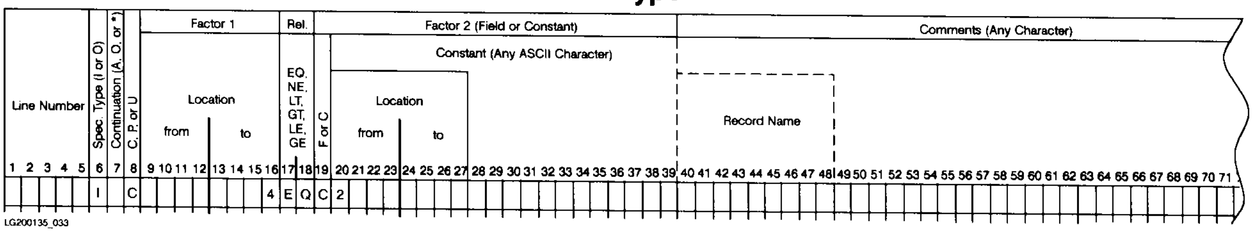
* If factor 2 is a constant, the length of the factor 1 field must not exceed 61 characters (see columns 20-80 for additional information).

** When factor 1 is packed, the field may actually represent 27 decimal digits and a sign.

Coding Rules

For readability these specifications should be right-justified. The entry which tells where the factor 1 field begins--where to read from--should end in column 12. The entry which tells where factor 1 ends--where to read through--should end in column 16. If you have factor 1 fields that are only one character long, you can leave columns 9-12 (from) blank and enter the single number of columns 13-16 (to). The example below describes the position of a factor 1 field which has a 2 entered in column 4.

Record Type



Columns 17-18: Type of Comparison

This entry tells XSORT what kind of comparison you want to make between factor 1 and factor 2. You will specify the type and location of factor 2 when you fill in columns 20-80 (if it is a constant) or 20-27 (if it is another field in the same record). When you specify Alternate Collating Sequence, XSORT modifies factors 1 and 2 before making comparisons.

Column 17-18 Entry	Meaning
EQ	Factor 1 must equal factor 2.
NE	Factor 1 must not equal factor 2.
LT	Factor 1 must be less than factor 2
GT	Factor 1 must be greater than factor 2.
LE	Factor 1 must be less than or equal to factor 2.
GE	Factor 1 must be greater than or equal to factor 2.

Column 19: Field or Constant

XSORT identifies records you want to sort by comparing the factor 1 field (specified in columns 9-16) with either a constant or another field in the same record. Column 19 tells the program whether it is a field or a constant that you wish to use in the comparison.

- C means a constant will be used.
- F means a field will be used.

When you put a C in column 19, you use columns 20-80 for the constant. When you put an F in column 19, you use columns 20-27 to identify the location of the factor 2 field in the record.

Columns 20-27: Factor 2 Field

The factor 2 field must be the same length as the factor 1 field and in the same record as the factor 1 field. These columns identify the location of the factor 2 field. Columns 20-23 (from) tell XSORT where the field starts. Columns 24-27 (to) tell the program where the field ends.

Coding Entries

For readability, both "from" and "to" sections should be right-justified. This is, the "from" entry should end in column 23, and the "to" entry should end in column 27.

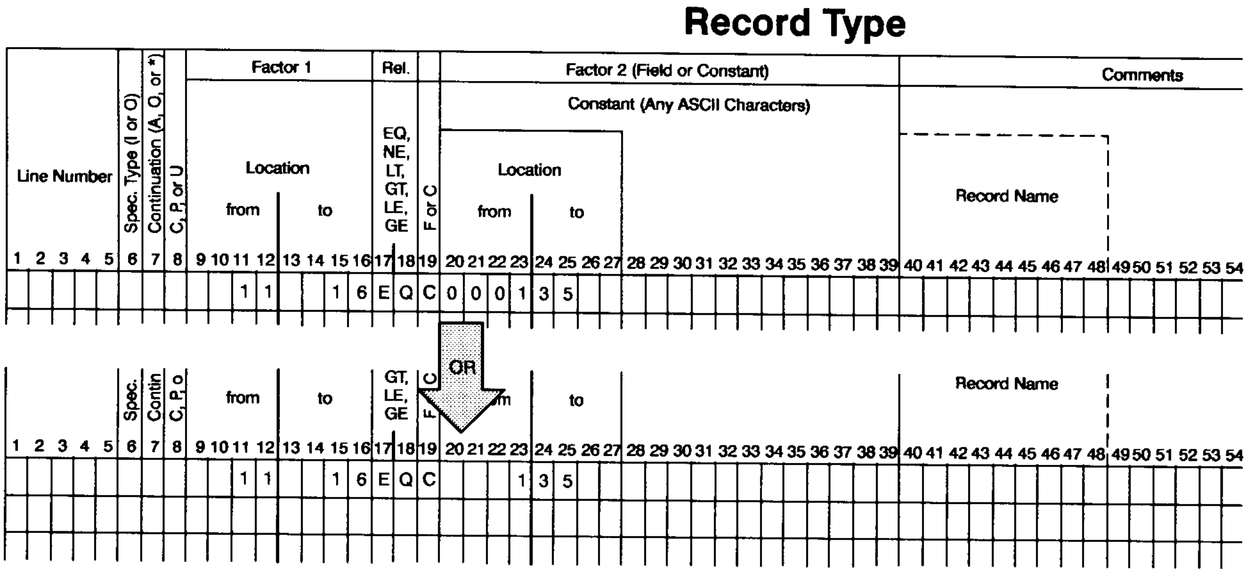
Sometimes fields are only a single character in length. When there is only one character, skip the 20-23 entry, leaving it blank. Enter the column number of the record position containing the character in columns 24-27.

Columns 20-80: Factor 2 Constant

If factor 2 is a constant, you use columns 20-80 to enter the constant you have selected. Normally, columns 40-80 are reserved for comments. They can, however, be used as necessary for constants longer than 20 characters or digits. The constant can be any arrangement of characters. When you enter a constant containing character data, placing a C in column 8, your entry must be the same length as the factor 1 field and must begin in column 20. Although the XSORT specification form includes 72 columns, your entries can extend to 80 columns.

Numeric Constants

Numeric constants are always right-justified according to their length. They must be the same length as the factor 1 field. If the factor 1 field is packed, the length of the factor 2 field will be twice as long. For example, if factor 1 was a six position field in the input record and factor 2 was the numeric constant 135, you would have to right justify your factor 2 entry to column 25. You would enter 135 in columns 23, 24, and 25. It would not be necessary to include leading zeros since XSORT reads blanks and zeros the same.



```
          11  16EQC000135
```

```
          11  16EQC   135
```

Unpacked Constants

Unpacked constants must be the same length as the factor 1 field. Suppose your factor 1 field takes up four positions. Your entry for factor 2 must do the same. If the factor 2 constant is the value 5, you would enter a 5 in column 23. You could leave 20, 21, and 22 blank or fill them with zeros. Negative unpacked constants require special attention. The negative sign must be "overpunched" on the last digit of the constant.

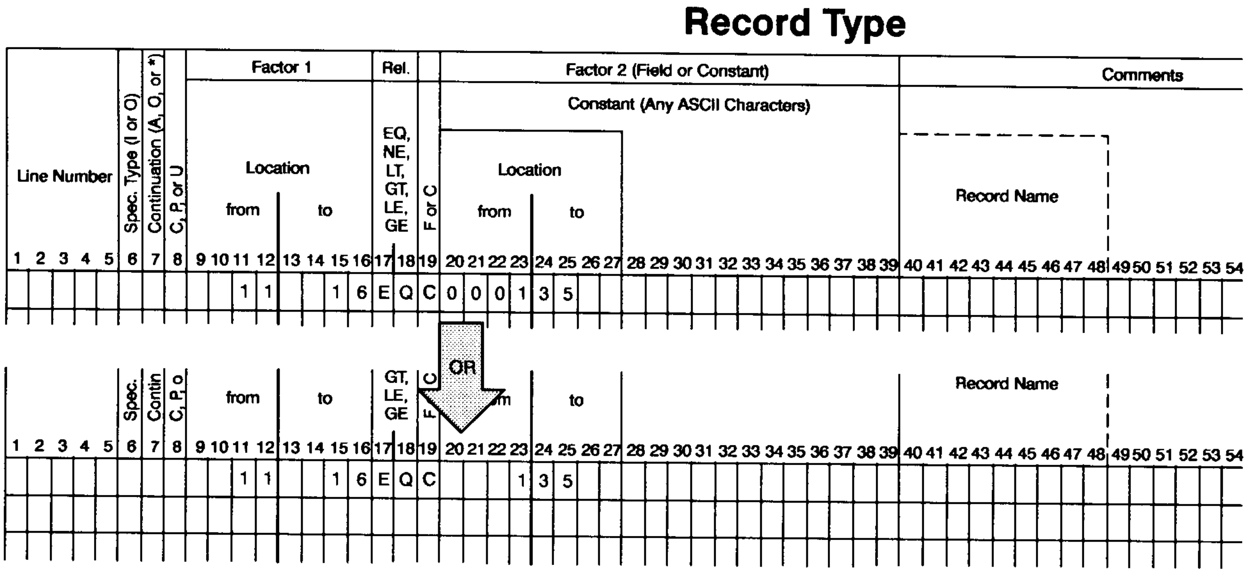
Coding Negative Constants (Unpacked)

If last digit in constant is:	You replace it with this character:
0	} (right brace)
1	J
2	K
3	L

If last digit in constant is:	You replace it with this character:
4	M
5	N
6	O
7	P
8	Q
9	R

Packed Constants

Packed numbers in this entry receive different treatment. If factor 1 contains a packed number, the length of the constant must be twice the length of the factor 1 field. This includes the sign, "+" or "-", which must be placed in the rightmost position of the constant after the last digit (negative signs cannot be "overpunched" on the last digit of the constant as they are for unpacked numbers). For example, here is the entry required to make a sort in which the record must contain a packed negative 3 (-3) in positions 3 and 4, an unpacked negative 24 (-24) in positions 10-13, and an unpacked negative 11 (-11) in positions 17-22.



```
          11  16EQC000135
          11  16EQC   135
```

Columns 40-80 Comments

Columns 40-80 are reserved for your comments. These comments will be printed along with your specifications if you instructed XSORT to do so. (Column 27 of your Header line must either be blank or contain a zero.) Some programmers find it convenient to write the names of records described in the Record Type Specifications here, so an area has been set aside in the comment section for that purpose.

Constants longer than 20 characters or digits will extend past column 40 into the comment area. They will be used correctly and not truncated at column 40. (See the discussion of factor 2 constants, columns 20-80.) Your comments do not influence the operation of your program in any way.

Chapter 6 Field Description Specifications

Field Description Specifications tell XSORT how to sort your records and format them in the output file.

The following table is an overview of field description specification column entries.

Columns	Entries	Explanation
1-5	Any value.	Line number.
6	F	Field Description line identification.
7*	N O F D *	Normal control field. Opposite control field. Forced control field. Data field. Comment line.
8*	P U C V	Packed decimal data. Unpacked decimal data. Character data. Force a data character into the data field
9-12	1-1000	Starting position of field in the record. May be left blank if the field is only one character long.
13-16	1-1000	End position of field in the record.
17	Any character	Forced control fields only (the character you want to change).
18	Any character.	Forced control or data fields only (the character you want to force into position).
19	blank Any character other than blank.	Forced control field line is not a continuation of the preceding line. Forced control line is a continuation of the preceding line.
20-39	blank	Reserved.
40-80	Any characters.	Comments.

* Columns that must be filled in.

Columns 1-5: Line Number

Line number is an optional entry. XSORT does not check your columns 1-5 entries for sequence. Your entry here is intended as a convenience, a record you may use for future reference.

Column 6: Line Type

An entry of F identifies this as a Field Description line.

Column 7: Field Type or Comments

The column 7 entry tells XSORT whether you are describing a control field, data field, or a comment line. If you are describing a control field, the entry tells the program how it is to be used. Entries in columns 7 and 8 interact and a separate column description section is included below to clarify this interchange of commands.

Column 7 Entry	Meaning To Program
N	This is a normal control field. The field will be sorted according to the sequence specified in column 18 of the Header Specifications.
O	This is an opposite control field. It tells the program to sort the data in the sequence opposite to that specified in column 18 of the Header Specifications.
F	This is a forced control field. The entries in columns 9-19 indicate how the control field is to be changed.
D	This is a data field (used for SORTR only).
*	This is a comment line.

Control Fields

(N, O, or F) Entries

When your file has more than one Record Section:

- The number of control fields do not have to be the same for all Record Sections.
- The total lengths of all the control fields do not have to be the same for all Record Sections (it is the maximum of these lengths that is entered in Header Specification columns 13-17).

Normal and Opposite Control Fields (N and O) Entry

These are fields used by the program to sort your input records. They are fields normally taken from the input records, although you can alter or add to them using forced control fields. (See later sections for more information concerning the use of forced control fields.)

Opposite Control Fields

At times you may wish to sort records so that some control fields are in ascending order and some are in descending order. You use opposite control fields to accomplish this.

Your Header Specification establishes the default sorting order. You will sort in ascending order if you have specified descending order in your Header Specification and called for an opposite control field. You will sort in descending order if your Header Specification called for ascending order and you enter O for an opposite control field. XSORT changes your control fields while building the work records if you specify opposite control fields. There is little reason to retain these fields and usually they are dropped -- by entering an X in column 28 of the Header line. If you wish to retain the original control fields in your output records in a meaningful form, repeat the information as a data field.

Sequencing Information with Control Fields

You determine the sequence of records in your Record-out sorts and the record addresses in your Address-out sorts by the order in which you describe control fields in your field description entries. How this occurs is illustrated below.

Input Record Position

Input Record Number	1	2	3	4	5	6	7	8
0		3	1				9	2
1		3	8			2	1	1
2		1	1			1	0	0
3		3	8			3	8	2
4		3	1			4	7	6
5		3	8			2	0	9
6		3	1			3	2	2

Item Number (positions 2–3) Number Ordered (positions 6–8)

Suppose your file is to be sorted in ascending order. (You have entered an A in column 18 of your Header line.) You have also specified that the file has a normal control field in positions 2-3. It has an opposite control field in positions 6-8.

The file, unsorted, might look like this.

This is how XSORT would sort the file:

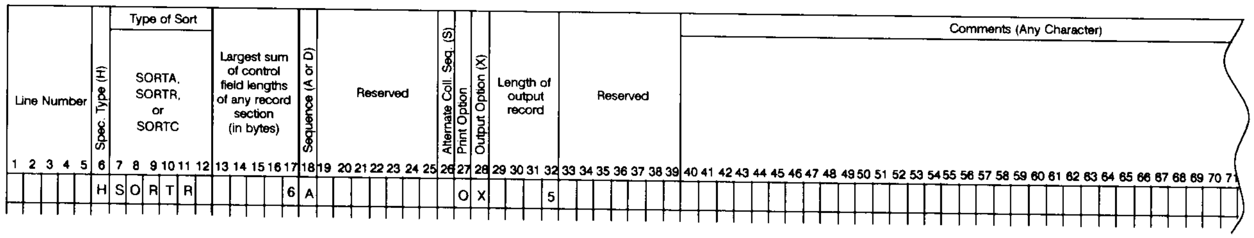
Output Record Position

Output Record Number	1	2	3	4	5
0	1	1	1	0	0
1	3	1	4	7	6
2	3	1	3	2	2
3	3	1		9	2
4	3	8	3	8	2
5	3	8	2	1	1
6	3	8	2	0	9

Item Number (positions 1–2) Number Ordered (positions 3–5)

XSORT uses the control field in columns 2-3 to sort the item numbers in ascending order. It then uses the second control field to sort out the number of units ordered in descending order within the units group.

Header



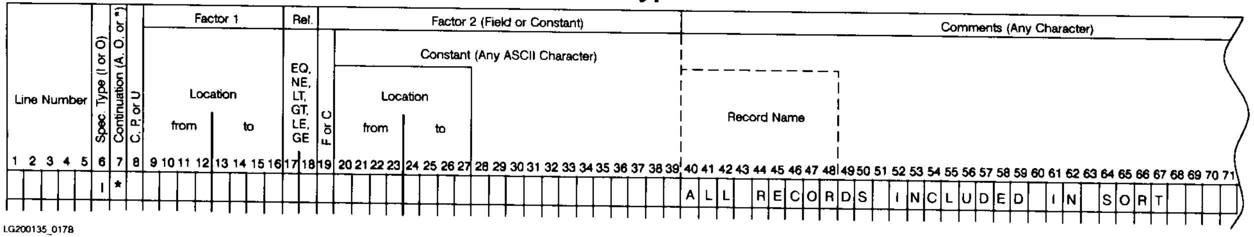
Line Number (1 2 3 4 5)	Spec. Type (H) (6)	Type of Sort: SORTA, SORTR, or SORTC (7 8 9 10 11 12)	Largest sum of control field lengths of any record section (in bytes) (13 14 15 16 17)	Sequence (A or D) (18)	Reserved (19 20 21 22 23 24 25)	Alternate Coll. Seq. (S) (26)	Print Option (27)	Output Option (X) (28)	Length of output record (29 30 31 32)	Reserved (33 34 35 36 37 38 39)	Comments (Any Character) (40–71)
	H	SORTR	6	A			O	X	5		

← Insert ALTSEQ specifications here (if used).

← Insert S OPTION specification here (if used).

LG200135_017A

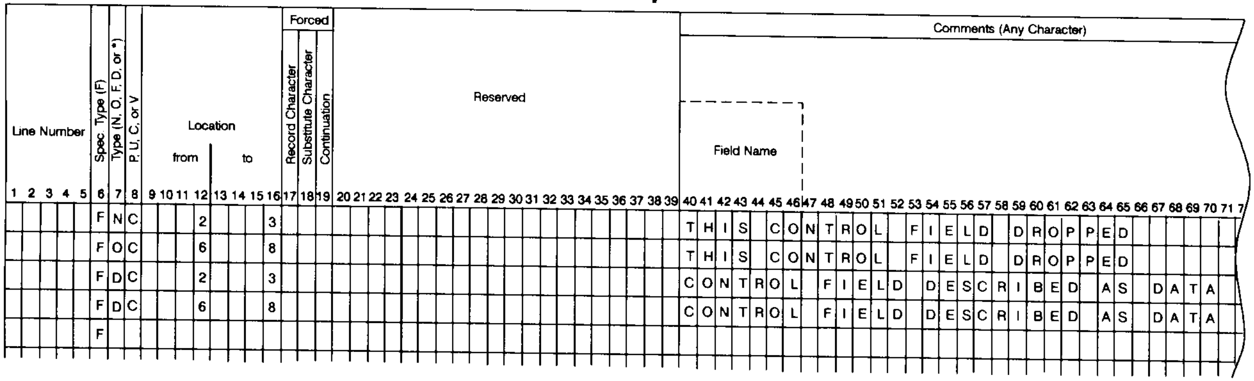
Record Type



Line Number (1 2 3 4 5)	Spec. Type (I or O) (6)	Continuation (A, O, or *) (7)	C, P, or U (8)	Factor 1 Location from (9 10 11 12)	Factor 1 Location to (13 14 15 16 17)	Rel. EQ, NE, LT, GT, LE, GE (17 18)	F or C (19)	Factor 2 (Field or Constant): Constant (Any ASCII Character) Location from (20 21 22 23)	Location to (24 25 26 27)	(28 29 30 31 32 33 34 35 36 37 38 39)	Comments (Any Character): Record Name (40–71)
	I	*									ALL RECORDS INCLUDED IN SORT

LG200135_017B

Field Description



Line Number (1 2 3 4 5)	Spec. Type (F) (6)	Type (N, O, F, D, or *) (7)	P, U, C, or V (8)	Location from (9 10 11 12)	Location to (13 14 15 16)	Forced: Record Character (17)	Forced: Substitute Character (18)	Continuation (19)	Reserved (20–39)	Comments (Any Character): Field Name (40–71)
	F	N	C	2	3					THIS CONTROL FIELD DROPPED
	F	O	C	6	8					THIS CONTROL FIELD DROPPED
	F	D	C	2	3					CONTROL FIELD DESCRIBED AS DATA
	F	D	C	6	8					CONTROL FIELD DESCRIBED AS DATA
	F									

LG200135_017C

Forced Control Fields

You can force a character into your control field by using forced control. There are special considerations you should keep in mind when you plan to use forced control fields.

- Only one character can be used in your forced control field.
- You can use either a conditional or an unconditional force.
- A force-all line must be preceded by one or more conditional force lines.

Conditional Force

A conditional force can be used only when a particular entry appears in the record. It allows you to change that entry to an alternate value for sorting purposes only. For example, you might have records to sort, each with a one-position control field. If that control field character were an A and you wished to change it to a $ for sorting, you would use conditional force. XSORT would respond to your instructions in the following way:

```
 * Build a work record from the input record.

             |------|---|------|                 |---|------|

    Input    | DATA | A | DATA |           Work  | A | DATA |
     Record    |------|---|------|                    Record  |---|------|
```

A 1 - position control field (A) in input record is moved to first control field position in work record.

* Change each A to a $.

```
          |---|-------|          |---|-------|
          | A |  Data |          | $ |  Data |
          |---|-------|          |---|-------|
          Original work record   Changed work record
```

There are two types of conditional forces:

- Conditional force on Normal or Opposite control field.
- Stand-alone conditional force.

Conditional force on Normal or Opposite Control Field

This allows you to specify a 1-character normal or opposite control field and have it converted to an alternate value on the work record, depending on its value on the input record. To use this option, you must first specify the normal or opposite control field and immediately follow it by the conditional force line or lines. Each conditional force line must show the same input field location in columns 13-16 and must include a continuation character in column 19 to indicate it is a continuation of the specification of the normal or opposite control field. This force option does not increase the length of your work record. It only overlays an already defined position.

Stand-Alone Conditional Force

If your conditional force lines are not a continuation of the specifications of a normal or opposite control field (see discussion above), XSORT will place a hex FF (if the sort is ascending) or a hex 00 (if the sort is descending) in the next available work record position. It will use that position as if it were a normal or opposite control field. Processing of the conditional force specifications then continues, based on that new position. Each conditional force line in a stand-alone group must show the same input field location in columns 13-16, even though that input field is never moved into the work record. (It is only used to determine what character, if any, is to overlay the hex FF or 00 in the work record.)

All lines except the first must include a continuation character in column 19. A stand-alone force group increases the length of your work record by 1.

Force-All

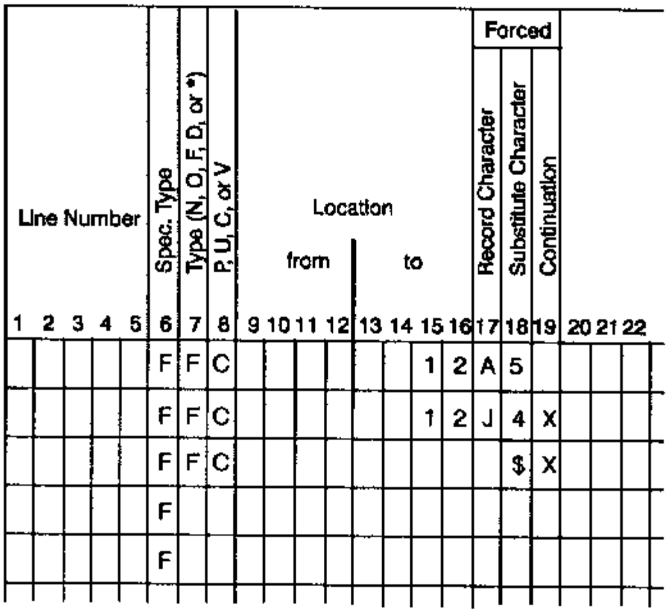
Force-all is a different type of conditional force. It works only when the control field of the input record does not contain a particular entry or entries. For this reason it is sometimes referred to as a "catch-all" operation.

A force-all line always follows a series of conditional force lines. The combined conditional and force-all abilities allow you to give XSORT instructions such as the following:

- If the control field is an A, place a 5 in the work record.
- If the control field is a J, place a 4 in the work record.
- If the control field contains neither an A nor a J, place a $ in the work record.

Here you are using two stand-alone conditional forces and following them with a force-all.

If the control field were in position 12 of the input records, here is how you would enter Field Description specifications for the example just described:



Line Number								Location								Forced					
					Spec. Type	Type (N, O, F, D, or *)	P, U, C, or V	from				to				Record Character	Substitute Character	Continuation			
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
					F	F	C							1	2	A	5				
					F	F	C							1	2	J	4	X			
					F	F	C										$	X			
					F																
					F																

Unconditional Force

Your unconditional force does not represent a field present in the input records. You might wish to place a 5 in the first position of every output record. Using an unconditional force control field, you could tell XSORT to place a 5 in the first available control field of the work record.

Control field portion	5		Data portion

All other control fields would then follow the 5 that has been unconditionally forced. Assume the input records are in this format.

Data	Control Field A	Data	Control Field B	Data

Your work record would look like this:

5	Control Field A	Control Field B	Data

Forced control fields affect work and output records only. They do not change input records. The column entries for columns 9-16, 17, 18, and 19 contain further information and examples concerning specifications for using forced control fields.

Data Fields (D) Entry

Data fields are fields you want the program to include in the output records but not use in sorting those records. The entry applies only to Record-out sorts. Within the Field Description lines of each Record Type section, control field lines must be placed before data field lines. When your file contains more than one Record Section:

- All Sections are not required to have the same number of data fields.
- Output data lengths do not have to be the same for all Sections. XSORT will place blanks to the right of shorter records to equalize lengths up to the output record length defined in Header Specifications columns 29-32.

Comment Lines (*) Entry

Comment lines have nothing to do with the way your program operates. They provide you with an opportunity to document your work. You can code documents anywhere within the XSORT specifications, but they will be printed only if a blank or 0 is in column 27 of your Header Specification.

Column 8: C, P, U, or V

Column 8 entries tell XSORT what type of data is to be used in the work records.

Column 8 Entry	Data Type Used	Maximum Field Length
C	Character equals one byte (8 bits) of data.	256 characters
P	Packed decimal data.	14 bytes (27 packed digits and sign)
U	Unpacked decimal data.	28 characters (unpacked digits)
V	Force a data character into the record.	1 character

You can force characters into your data field by placing a V in column 8 and specifying the character you want forced into that position in column 18. The character will then be placed in the next available data field position of the work record.

NOTE Column 8 entries in the Field Description Specifications may seem similar to the column 8 entries you entered as Record Type descriptions. They should not be confused. Column 8 of the Record Type specifications helps select the records you wish to sort. Column 8 of the Field Description tells how you want data used in sorting and outputting your work records.

Packed Control Fields (Normal or Opposite)

XSORT also changes the control fields while building its work record if you specify packed control fields. The X in column 28 of your Header Specification will drop these changed control fields on output. If you wish to retain them in meaningful form they should be entered again as data fields.

Columns 7 and 8 - Combined Entries		
Column 7	**Column 8**	**Maximum Field\Length**
N or O	P	14
N or O	U	28
N or O	C	256
F	C1	1
D	P	14
D	U	28
D	C	256
D	V	1
*		Comment line

NOTE For an unconditional force and a force-all line, column 8 must contain a C.

Columns 9-16: Field Locations

Columns 9-16 locate the input record positions that contain the field. The starting position (from) is identified in columns 9-12. The end position (to) is identified in columns 13-16. The order you impose in your Field Description Specifications will determine the order in which they will be located in the output records. For example, suppose your input records look like this.

Input Record

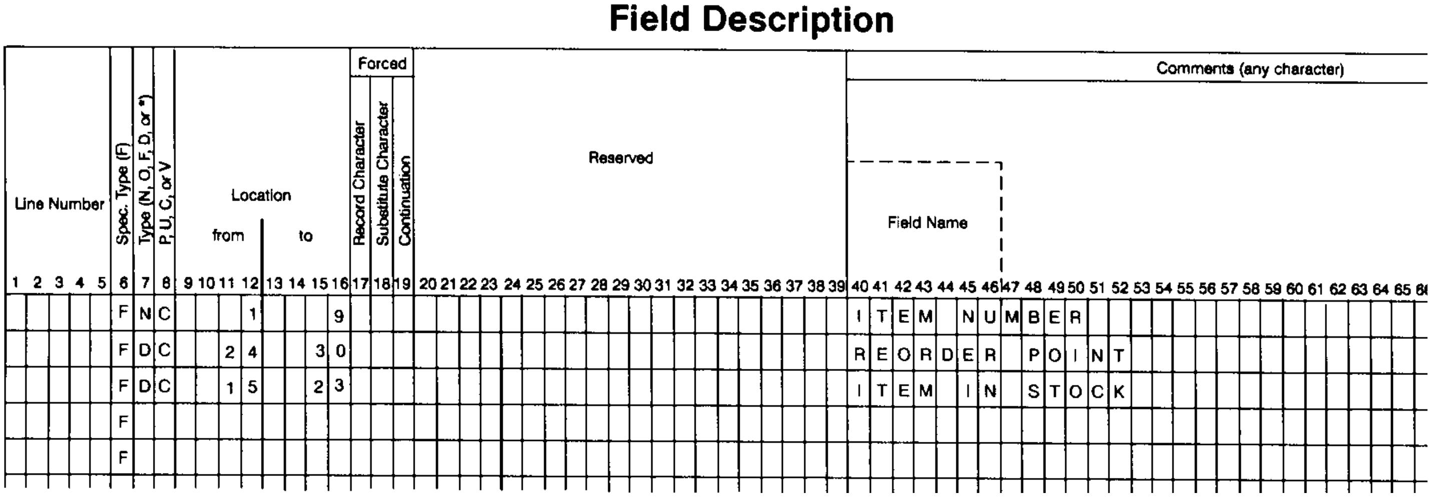
Field name	ITEM	PRICE	ITEMS IN STOCK	REORDER POINT
record position	1 9	10 14	15 23	24 30

And this is the way you want your output record to look.

Output Record

Field name 2205	ITEM	REORDER POINT	ITEMS IN STOCK
record position	1 9	10 16	17 25

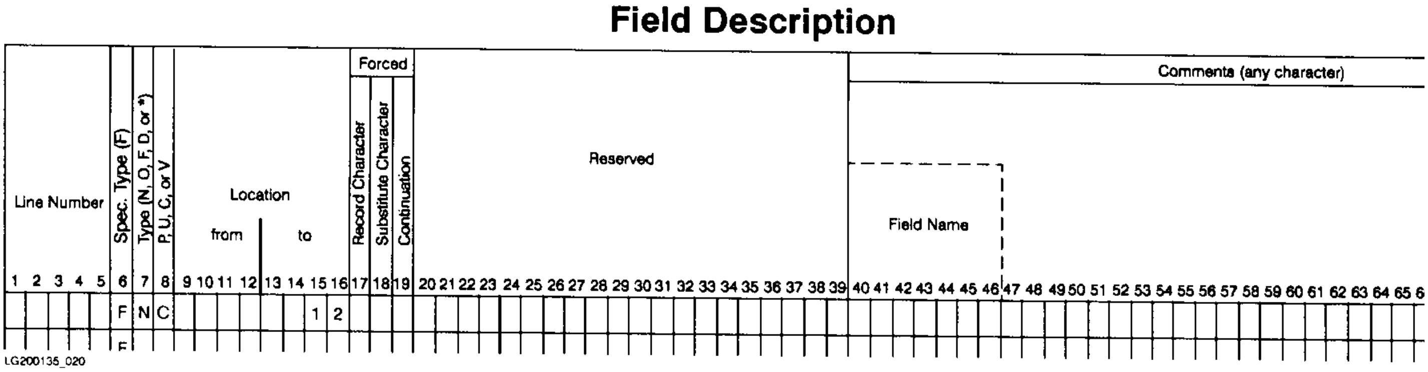
If you wanted to sort your records by item number, this is the way you would enter your Field Description Specifications.



Field Description

Line Number (1-5)	Spec. Type (F) (6)	Type (N, O, F, D, or *) (7)	P, U, C, or V (8)	Location from (9-12)	Location to (13-16)	Forced: Record Character (17)	Forced: Substitute Character (18)	Forced: Continuation (19)	Reserved (20-39)	Comments (any character): Field Name (40-)
	F	N	C	1	9					ITEM NUMBER
	F	D	C	24	30					REORDER POINT
	F	D	C	15	23					ITEM IN STOCK
	F									
	F									

When you have a one-character control field, the procedure is the same with one exception. When the fields are only one character long, you leave columns 9-12 (from) blank and enter the number of the record position containing the character in columns 13-16. As an example, suppose you wanted the 1-character field in position 12 of your input record to be in the first position of the sorted output record. You would describe the field in the first line of your Field Description Specifications, entering it as below.



Field Description

Line Number
Spec. Type (F)
Type (N, O, F, D, or *)
P, U, C, or V
Location
from
to
Forced
Record Character
Substitute Character
Continuation
Reserved
Comments (any character)
Field Name
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 6
LG200135_020

F N C 1 2

F

Right-justifying entries is recommended. The starting (from) position of the input records ends in column 12. The end (to) position ends in column 16. In the example above, because it is a 1-character field, the 9-12 positions are left blank and the entry of the character's position is right-justified in column 16.

Field Length

The maximum allowable length of the field depends on your column 8 entry.

COL. 8 ENTRY	MAXIMUM FIELD LENGTH
P	14 characters
U	28 characters
C	256 characters
V	1 character

Column 17: Conditionally Forced Characters

Forced control fields were generally discussed under the description of column 7 entries. In column 17 you make an entry only when you wish to use a conditional force; for example, if you wanted to place a $ in a control field presently containing an A, conditional force does not change the input record but it will replace each A with a $ in that position in the work and output records.

Your entry in column 17 identifies the character that you want replaced in the control field. XSORT checks the input records to see if the control field contains the character you want replaced. Your entry in column 18 will then give XSORT the character you wish to have forced into that control field's position in the work record.

Sometimes control fields contain any one of several characters and you may wish to force a change for each of them. If this is the case, use a continuation character in column 19 (discussed below) on the specifications for all but the first of the forced characters.

Column 18: Forced Character

The discussion of column 7 entries covers the subject of forced characters in detail. You use your column 18 entry only when you are using forced control fields to sort your records or when you are using forced data fields. In forcing a control field, the entry in column 18 either replaces the character you specified in column 17 or adds a character to the work record. If you are using the entry to force a data field, the character in column 18 will be added in the next available data field position of the work record.

NOTE You can only use forced characters with 1-character fields, and your input records are not changed when you use forced characters.

Column 19: Continuation

A non-blank character here says that this line is a continuation of the force character specifications group begun on the preceding line or lines.

When several force character specifications apply to the same 1-character control field, they must be entered together as a group. The first line of the group must have a blank in column 19 and all subsequent lines must have a non-blank character in column 19 to indicate continuation of the group of specifications.

Summarizing Force Character Entries

Conditional Force Character

1. Fill in columns 1-6 as you would for any control field.
2. Enter an F in column 7.
3. Enter C in column 8.
4. Leave columns 9-12 blank.
5. Specify the position of the control field in the input record in columns 13-16.
6. Enter the character you want replaced in column 17.
7. Enter your replacement character in column 18. Any character may be used.
8. Place any character in column 19 if this is the 2nd or more conditional force on the same control field. This will tell the program that this is a continuation of the preceding line.

Force-all Character

1. Fill in columns 1-6 as you would for any control field.
2. Enter an F in column 7.
3. Enter C in column 8.
4. Leave columns 9-17 blank.
5. Put the replacement character for the control field in column 18.
6. Place any character in column 19. This will tell the program that this is a continuation of the preceding line.

Unconditional Force Character

1. Fill in columns 1-6 as you would for any control field.
2. Enter an F in column 7.
3. Enter a C in column 8.
4. Leave columns 9-17 blank.
5. Put the character you are forcing in column 18.
6. Leave column 19 blank.

Examples Using Forced Control Characters

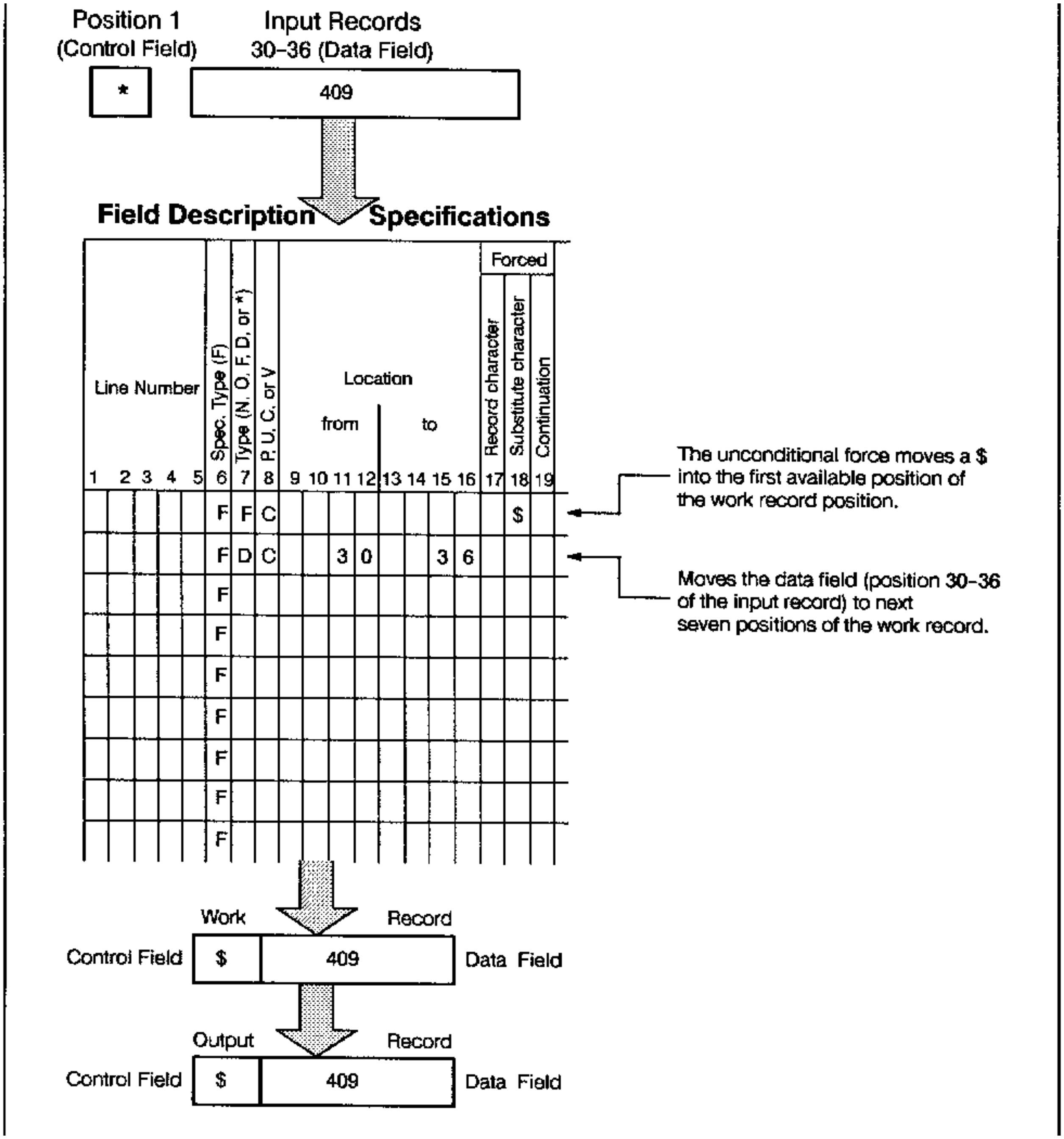
Forced control characters can be used to add to or replace a character in a one-position control field. Examples below show how to code four types of forced control characters:

- Unconditional force.

- Force-all.
- Conditional force on normal or opposite control fields.
- Stand-alone conditional force.

Example 1: Unconditionally Forced Character

In this example a control field ($) is unconditionally forced into the work and output records.



Line Number 1-5	Spec. Type (F) 6	Type (N, O, F, D, or *) 7	P, U, C, or V 8	Location from 9-12	Location to 13-16	Forced: Record character 17	Forced: Substitute character 18	Forced: Continuation 19
	F	F	C				$	
	F	D	C	30	36			
	F							
	F							
	F							
	F							
	F							
	F							
	F							

The unconditional force moves a $ into the first available position of the work record position.

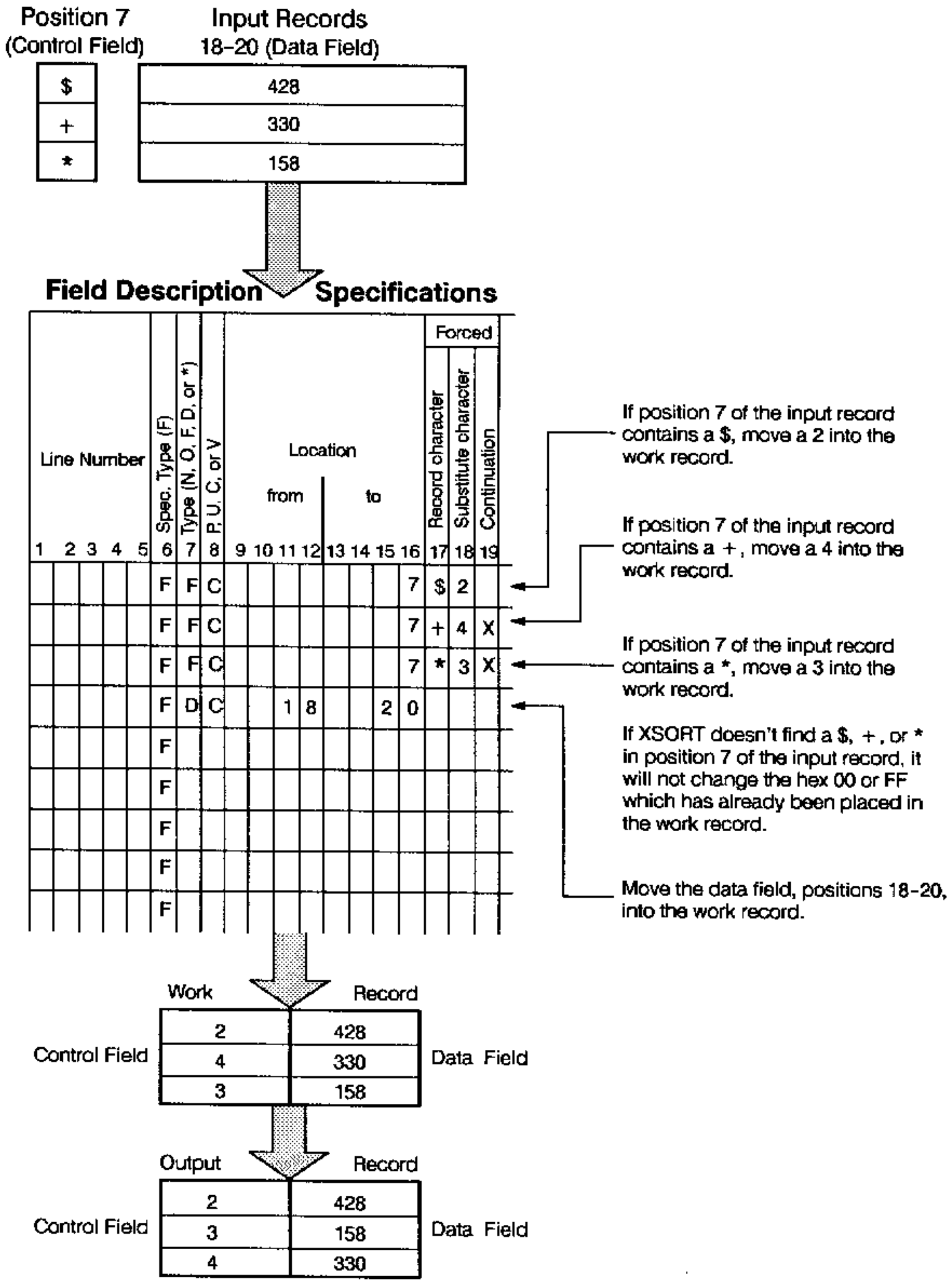
Moves the data field (position 30-36 of the input record) to next seven positions of the work record.

Example 2: Stand-Alone Conditional Force

This example shows how different control field characters can be forced into position in the work record, depending on the character present in an input record position.

Here the column 19 entry is used to specify conditional forcing of more than one character. Because continuation lines are used in this example, only one position in the work and output records is defined by the first three lines. If column 19 were left blank and no continuation lines were used, each line would then define a new position in the work and output records.

When you sort records in ascending order, having placed an A in column 18 of the Header, XSORT places hex FF into the work record before you force any characters. When you sort records in descending order, having placed a D in column 18 of the Header, the program places hex 00 into the work record before any characters are forced.



Position 7 (Control Field)	Input Records 18-20 (Data Field)
$	428
+	330
*	158

Line Number 1 2 3 4 5	Spec. Type (F) 6	Type (N, O, F, D, or *) 7	P, U, C, or V 8	Location from 9 10 11 12	Location to 13 14 15 16	Forced Record character 17	Forced Substitute character 18	Forced Continuation 19
	F	F	C		7	$	2	
	F	F	C		7	+	4	X
	F	F	C		7	*	3	X
	F	D	C	18	20			
	F							
	F							
	F							
	F							
	F							

If position 7 of the input record contains a $, move a 2 into the work record.

If position 7 of the input record contains a +, move a 4 into the work record.

If position 7 of the input record contains a *, move a 3 into the work record.

If XSORT doesn't find a $, +, or * in position 7 of the input record, it will not change the hex 00 or FF which has already been placed in the work record.

Move the data field, positions 18-20, into the work record.

Work Record

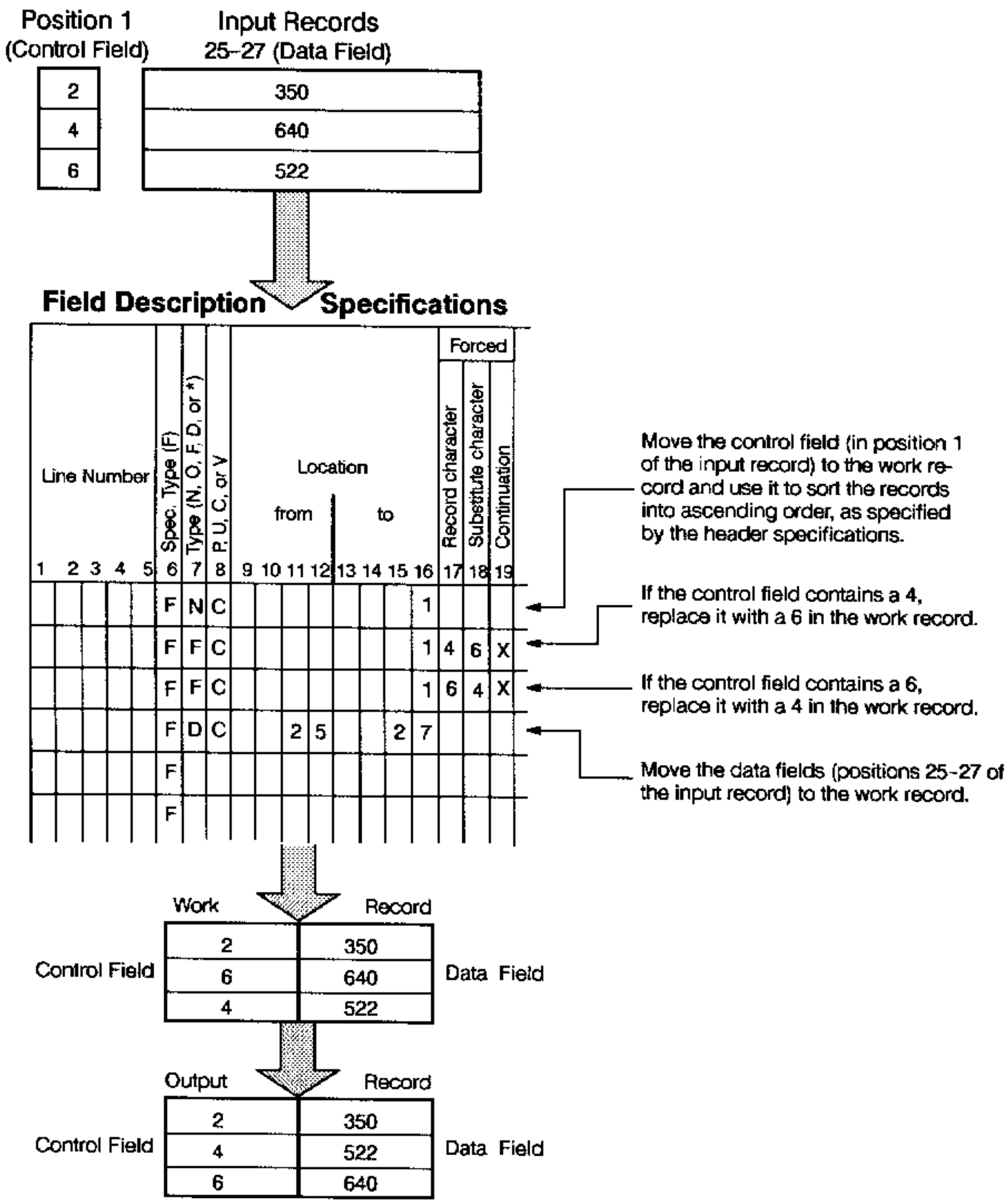
Control Field	Data Field
2	428
4	330
3	158

Output Record

Control Field	Data Field
2	428
3	158
4	330

Example 3: Conditional Force on Normal or Opposite Control Field

In this example, as in the one previous, the control field in the input record changes the work and output records. This example is significantly different, however. Here the control field is moved to the work record and then changes are made. This occurs because the first control fields are normal control fields.



Position 1 (Control Field)	Input Records 25-27 (Data Field)
2	350
4	640
6	522

Field Description Specifications

Line Number					Spec. Type (F)	Type (N, O, F, D, or *)	P, U, C, or V	Location								Forced		
								from				to				Record character	Substitute character	Continuation
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
					F	N	C								1			
					F	F	C								1	4	6	X
					F	F	C								1	6	4	X
					F	D	C			2	5			2	7			
					F													
					F													

Move the control field (in position 1 of the input record) to the work record and use it to sort the records into ascending order, as specified by the header specifications.

If the control field contains a 4, replace it with a 6 in the work record.

If the control field contains a 6, replace it with a 4 in the work record.

Move the data fields (positions 25-27 of the input record) to the work record.

Work Record

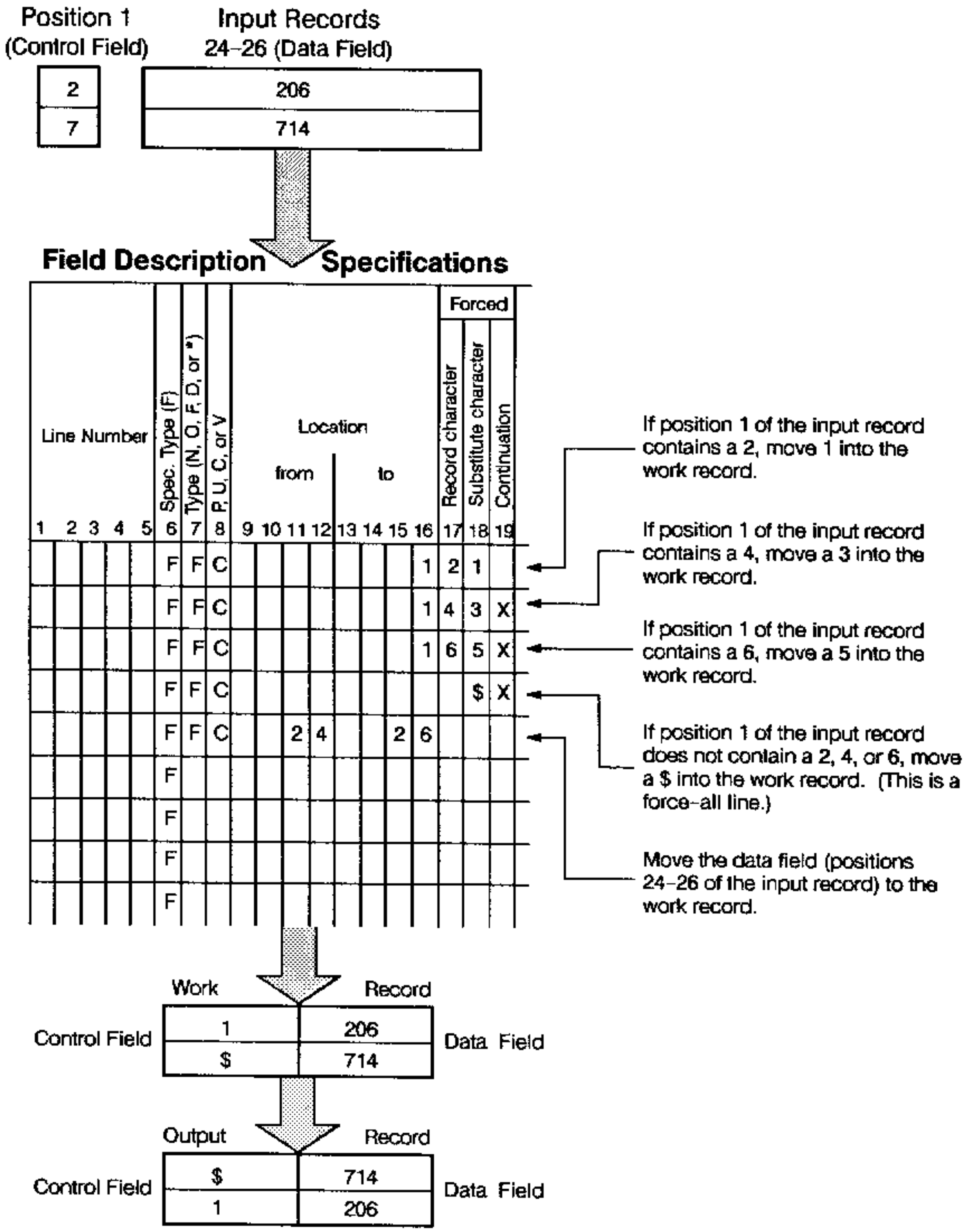
Control Field	Data Field
2	350
6	640
4	522

Output Record

Control Field	Data Field
2	350
4	522
6	640

Example 4: Force-All

In the example, if control fields contain characters other than 2, 4, and 6, use a force-all line so that you won't have a hex FF or 00 present in the work and output records.



Line Number 1	2	3	4	5	Spec. Type (F) 6	Type (N, O, F, D, or *) 7	P, U, C, or V 8	Location from 9	10	11	12	to 13	14	15	16	Forced Record character 17	Substitute character 18	Continuation 19
					F	F	C								1	2	1	
					F	F	C								1	4	3	X
					F	F	C								1	6	5	X
					F	F	C										$	X
					F	F	C			2	4			2	6			
					F													
					F													
					F													
					F													

If position 1 of the input record contains a 2, move 1 into the work record.

If position 1 of the input record contains a 4, move a 3 into the work record.

If position 1 of the input record contains a 6, move a 5 into the work record.

If position 1 of the input record does not contain a 2, 4, or 6, move a $ into the work record. (This is a force-all line.)

Move the data field (positions 24-26 of the input record) to the work record.

Columns 20-39

Reserved for future use.

Columns 40-80 Comments

These columns are reserved for your comments. Columns 40-45 are set aside by dotted lines as a convenient place to write down the names of fields described in the Field Description lines. If you specified in your Header line that you wanted to have your specifications printed, your comments will be printed also. These comments have no effect on your program. Reserved for future use.

7 Chapter 7 Using XSORT Commands

Introduction

This section contains examples of the MPE commands necessary to run XSORT in interactive and job stream modes. These examples also encompass the various input and output file options provided by the utility.

Running XSORT

Extra Function Sort for RPG can be run in job stream or interactively. Streaming is the recommended method because the program does not presently prompt for input.

On the following pages are examples of the MPE commands needed to run the program in several modes. The grid below indicates the alternatives open to you. The vertical column on the left presents the ways in which the specifications can be entered--either in job streams with specifications imbedded (or using the formal file designator, XSORTEXT); or in sessions in which the data is entered interactively (or again, using XSORTEXT). In the examples, the file equation XSORTEXT=$STDIN has been left out where it would otherwise be used as that is the default setting for XSORTEXT.

The horizontal columns at the top of the grid show the choices open to you concerning input and output files as well as the Count-only sort (SORTC) designated Example #9.

Each box in the grid contains the number of the example (or combination of examples) to refer to for the sort to be run. To use the grid, you must determine what kind of sort you will require (whether it is an in place sort or one using multiple input files, for example). Then you must determine your method of running the program (in job stream using XSORTEXT, for example). You can find the means by which this work can be successfully accomplished in the pages that follow and, if your needs are not explicitly covered, it is possible to combine the information included in the examples to enter your requirements into the system.

	Single file input to different output (doesn't already exist)	Single file input -- inplace	Single file input to different output already exist)	Multiple input	SORTC (count-only)*
Job Stream: Specs Imbedded	#1	#6	#7	#8	#9
Job Stream: XSORTEXT	#2	(Combine) #2 & #6	(Combine) #2 & #7	(Combine) #2 & #8	(Combine) #2 & #9
Session: Specs Interactive	#3	(Combine) #3 & #6	(Combine) #3 & #7	(Combine) #3 & #8	(Combine) #3 & #9
Session: XSORTEXT (Case 1)	#4	(Combine) #4 & #6	(Combine) #4 & #7	(Combine) #4 & #8	(Combine) #4 & #9
Session: XSORTEXT (Case 2)	#5	(Combine) #5 & #6	(Combine) #5 & #7	(Combine) #5 & #8	(Combine) #5 & #9

Case 1 - XSORTIN-and XSORTOUT are disk files.

Case 2 - XSORTIN=$STDIN and XSORTOUT=$STDLIST.

* Example #9 uses single file input. Count-only (SORTC) can also be used with multiple file input. To determine the appropriate commands for a multiple file SORTC, see Example #8 regarding how it specifies multiple input files (XSORTIN1...XSORTIN9).

About the Sample Sort

A very simple sort was devised for these examples. The input file, named TSTIN, consists of five records.

```
AB123GSD212345   JONES          X
AB234CICO12345   ALBERS
AC102GSD100001   WOODSON
AC321ABCD10001   FRANKEN
BB222CICO12345   WOODS
```

The specifications for most of the examples, whether entered by the user or supplied by the file, TSTSPEC, are these:

```
00000HSORTR     4A         X  33
00005I C   33NECX
00010FNC   6    9
00020FDC   1   32
00030FDV              *
```

In examples 5 and 8, it is necessary to include an S Option specification for reasons which are explained later. A file, TSTSPEX, which includes the S Option specs, was created for these examples. The entry in columns 10-16, which limits the "maximum number of records to be sorted", is set for Example 8.

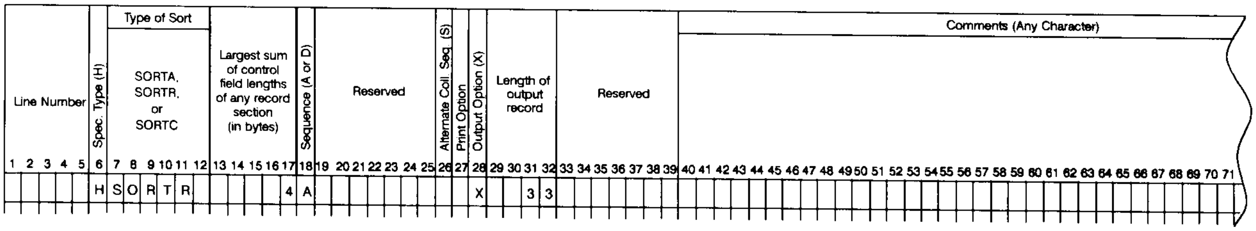
```
00000HSORTR     4A         X  33
00000S Y   35
00005I C   33NECX
00010FNC    6   9
00020FDC    1  32
00030FDV              *
```

In Example 7, a file named TSTFILE, which exists previous to the sort, is overlaid by the sorted file. TSTFILE does not appear in the example as printed.

Sort Specifications

The specifications for our sort, if entered in the XSORT specification forms, would look like this:

Header



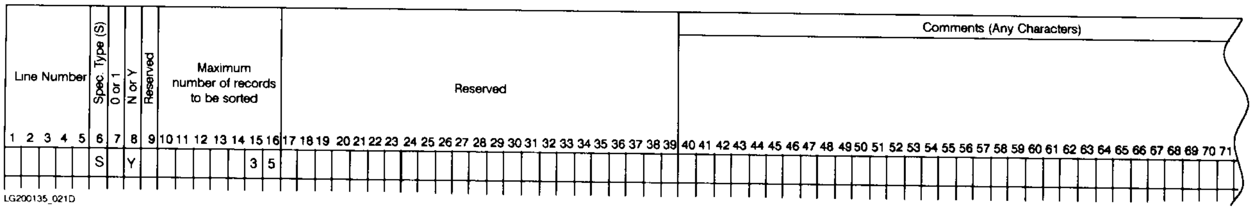
Line Number (1-5)	Spec. Type (H) (6)	Type of Sort: SORTA, SORTR, or SORTC (7-12)	Largest sum of control field lengths of any record section (in bytes) (13-17)	Sequence (A or D) (18)	Reserved (19-25)	Alternate Coll. Seq. (S) (26)	Print Option (27)	Output Option (X) (28)	Length of output record (29-32)	Reserved (33-39)	Comments (Any Character) (40-71)
	H	SORTR	4	A				X	33		

Insert ALTSEQ specifications here (if used).

Insert S OPTION specification here (if used).

LG200135_021A

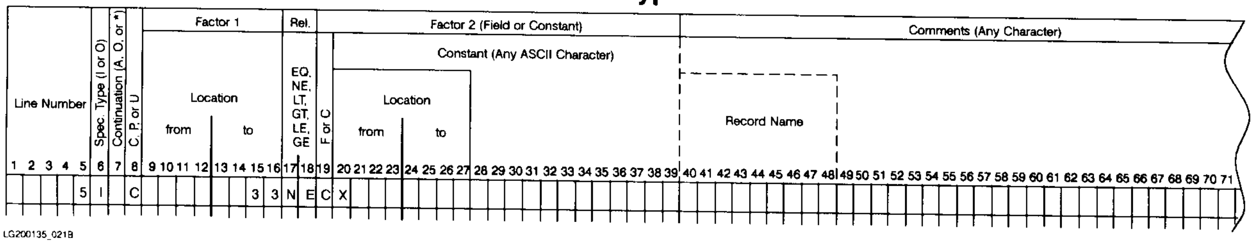
S Option



Line Number (1-5)	Spec. Type (S) (6)	0 or 1 (7)	N or Y (8)	Reserved (9)	Maximum number of records to be sorted (10-16)	Reserved (17-39)	Comments (Any Characters) (40-71)
	S		Y		35		

LG200135_021D

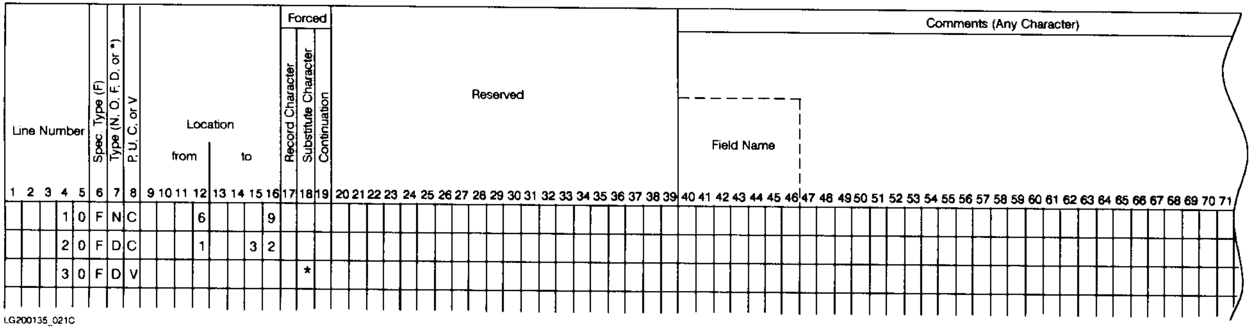
Record Type



Line Number (1-5)	Spec. Type (I or O) (6)	Continuation (A, O, or *) (7)	C, P, or U (8)	Factor 1 Location from (9-12)	Factor 1 Location to (13-16)	Rel. EQ, NE, LT, GT, LE, GE (17-18)	F or C (19)	Factor 2 (Field or Constant): Constant (Any ASCII Character) / Location from (20-23)	Location to (24-27)	(28-39)	Comments (Any Character): Record Name (40-47)	(48-71)
5	I		C		33	NE	C	X				

LG200135_021B

Field Description



Line Number (1-5)	Spec. Type (F) (6)	Type (N, O, F, D, or *) (7)	P, U, C, or V (8)	Location from (9-12)	Location to (13-16)	Forced: Record Character (17)	Forced: Substitute Character (18)	Continuation (19)	Reserved (20-39)	Comments (Any Character): Field Name (40-46)	(47-71)
10	F	N	C	6	9						
20	F	D	C	1	32						
30	F	D	V				*				

LG200135_021C

A brief explanation of each specification follows.

Header Specifications

Column	Entry	Explanation
1-5	blank	No line numbers are required.
6	H	The H designates this as the Header specification for the sort.
7-12	SORTR	SORTR indicates that this is a Record-out sort.
13-17	4	Largest sum of control field lengths for any record section, in bytes. In this sort the total is 4.
18	A	Records are to be sorted in ascending order by control field.
28	X	Control fields are to be dropped on output in this sort.
29-32	33	33 is the length of the output records in this sort. It includes only the length of the data fields since the control fields are being dropped.

S Option Specifications

Column	Entry	Explanation
1-5	blank	No line numbers are required.
6	S	Indicates that this is the S Option specification for the sort.
8	Y	XSORTOUT may already exist and be a different file than XSORTIN.
10-16	35	Maximum number of records to be sorted.

Record Type Specifications

Column	**Entry**	**Explanation**
1-5	5	Line number (optional).
6	I	This is an include line. All records described will be included in the sort.
8	C	Use full characters
13-16	33	The factor 1 field, which is only one character long, ends at this position.
17-18	NE	Factor 1 must not equal factor 2.
19	C	Factor 1 is a constant.
20	X	This is the factor 2 constant.

Field Description Specifications

Column	**Entry**	**Explanation**
1-5	10,20,30	Line numbers (optional).

Column	Entry	Explanation
6	F	These are field description lines.
7	N	This is a normal control field. This is a data field.
	D	This is a data field.
8	C	Use full characters in the field. Force a data character into the data field.
9-12	V	Force a data character into the data field
13-16	9,32	These are the starting positions of the data fields in the records.
18	*	This is the character you are forcing by using your column 8 entry.

About the Job Stream Examples

The easiest way to prepare job control commands is to enter them in a disk file using EDIT/3000. When you want to run a job stored in a disk file, you can use the MPE :STREAM command. All of the stream jobs described in the examples use this method.

When preparing a job to be streamed, you must substitute another character for the colon normally used with MPE commands. Any ASCII character may be used except the letters A through Z or the numbers 0 through 9. The character most commonly used is the exclamation point.

Although each of the job stream examples has unique characteristics, all have some in common. A general discussion of the commands for Example 1 will cover most of the common ground for the various sort configurations covered here. Where unique qualities are present, additional explanations will be appended to the example.

Example 1: Job Stream With Specifications Imbedded

Single input to different output.

```
1       !JOB TSTXSRT1,MGR.SUBSYS,XSORT;OUTCLASS=,1

2       !COMMENT  *************************

       !COMMENT  *************************

       !COMMENT THIS IS AN EXAMPLE OF HOW TO USE "XSORT" WITH

       !COMMENT SINGLE INPUT-TO-DIFFERENT OUTPUT

3       !FCOPY FROM=TSTIN;TO=

4       !PURGE TSTOUT

5       !FILE XSORTIN=TSTIN

6       !FILE XSORTOUT=TSTOUT

7       !RUN XSORT.PUB.SYS

8       0000HSORTR     4A         X 33
```

```
      00005I C      33NECX

      00010FNC   6   9

      00020FDC   1  32

      00030FDV         *

9      !EOD

10     !SAVE TSTOUT

      !FCOPY FROM=TSTOUT;TO=

11     !EOJ
```

1. The first command in a job must be the JOB command, followed by the user name, account name, and optionally the group name. In production work it is useful to specify a job name before the user name to identify the specific job that is being run. In this example, TSTXSRT1 is the job name.
2. Comment lines are used to describe the function or nature of the job.
3. The first program to be run in the job is FCOPY. It is included in order to show the contents of the input file prior to sorting. FCOPY is instructed to copy from the file TSTIN to the standard job/session device, $STDLIST.
4. A file named TSTOUT is used for each of the jobs. It is purged initially to make sure that the name does not exist in the group.
5. The file referenced as XSORTIN is equated to the actual file named TSTIN.
6. The file referenced as XSORTOUT is equated to the actual file named TSTOUT.
7. The program XSORT.PUB.SYS is run.
8. These are the specifications for the sort. Note that they are not entered with the exclamation point since they are used as data by the XSORT program, instead of being used as MPE commands.
9. !EOD, entered with an exclamation point, indicates that you have entered all necessary data for the sort.
10. The output file TSTOUT is to be saved.
11. !EOJ signifies that you have completed the job entries.

The following is a physically condensed copy of the previous example after it was run. It includes the output of the sort.

```
:JOB TSTXSRT1,MGR.SUBSYS,XSORT
PRIORITY = DS; INPRI = 8; TIME = UNLIMITED SECONDS
JOB NUMBER =   #J59
WED, MAR  11,  1981,   9:44.AM
HP3000 /  MPE  IV C.00.P8
D U K E
COMMENT   *************************
:COMMENT   *************************
:COMMENT  THIS IS AN EXAMPLE OF NOW TO USE "XSORT" WITH
:COMMENT  SINGLE INPUT-TO-DIFFERENT OUTPUT
:FCOPY FROM=TSTIN;TO=
HEWLETT-PACKARD 32212A.3.13 FILE COPIER WED, MAR 11, 1981, 9:44 AM
(C)-HEWLETT-PACKARD CO. 1980

*200*
WARNING:  FROMFILE RECSIZE IS 72 BYTES, TOFILE RECSIZE IS 133 BYTES.

AB123GSD212345      JONES           X
AB234CIC012345      ALBERS
AC102GSD100001      WOODSON
AC321ABCD10001      FRANKEN
BB222CIC012345      WOODS
EOF FOUND IN FROMFILE AFTER RECORD 4
5 RECORDS PROCESSED *** 0 ERRORS
&empty;
END OF SUBSYSTEM
:PURGE TSTOUT
:FILE XSORTIN=TSTIN
:FILE XSORTOUT=TSTOUT
:RUN XSORT.PUB.SYS

HP32104A.05.00      EXTRA  FUNCTION SORT FOR RPG/3000       (XSORT)
(C) HEWLETT-PACKARD  CO. 1980     WED, MAR 11, 1981,   9:44 AM

00000HSORTR    4A           X  33
00005I C     33NECX
00010FNC  6   9
00020FDC  1  32
00030FDV        *

RECORDS READ = 5
RECORDS SELECTED & SORTED = 4
  NUMBER OF COMPARES = 7
  SCRATCHFILE I/O = 2
  CPU SECONDS = .335
  ELAPSED SECONDS = 2.992
TOTAL CPU SECONDS = 1.006

END OF PROGRAM
:SAVE TSTOUT
:FCOPY FROM=TSTOUT;TO=

HEWLETT-PACKARD 32212A.3.13 FILE COPIER      WED, MAR 11, 1981,     9:45 AM
(C) HEWLETT-PACKARD CO. 1980

*200*
WARNING:  FROMFILE RECSIZE IS 33 BYTES, TOFILE RECSIZE IS 133 BYTES.

AC321ABCD10001      FRANKEN     *
```

```
AB234CIC012345      ALBERS      *
BB222CIC012345      WOODS       *
AC102GSD100001      WOODSON     *
EOF FOUND  1N  FROMFILE AFTER  RECORD 3
4 RECORDS PROCESSED *** 0 ERRORS

END OF SUBSYSTEM
:EOJ
CPU SEC. = 6.  ELAPSED MIN. = 1.  WED, MAR 11, 1981,    9:45 AM
```

Example 2: Job Stream with XSORTEXT.

Single input to different output.

```
!JOB TSTXSRT2,MGR.SUBSYS,XSORT;OUTCLASS=,1
!COMMENT THIS IS AN EXAMPLE OF A SINGLE INPUT-TO-
!COMMENT DIFFERENT-OUTPUT SORT.THE JOB IS STREAMED,
!COMMENT THE SPECIFICATIONS ON XSORTEXT.
!COMMENT******************************************
!COMMENT******************************************
!FCOPY FROM=TSTIN;TO=
!PURGE TSTOUT
!FILE XSORTIN=TSTIN
!FILE XSORTOUT=TSTOUT
!FILE XSORTEXT=TSTSPEC
!RUN XSORT.PUB.SYS
!SAVE TSTOUT
!FCOPY FROM=TSTOUT;TO=
!EOJ
```

NOTE The only difference between Example 1 and Example 2 is the use of TSTSPEC in the equation using XSORTEXT. TSTSPEC is an EDITOR file containing the same XSORT specs as those imbedded in the previous job stream.

Example 3: Session With the Specifications Entered Interactively.

Single input to different output.

```
:PURGE TSTOUT

:FILE XSORTIN=TSTIN

:FILE XSORTOUT=TSTOUT

:RUN XSORT.PUB.SYS

HP32104A.05.00     EXTRA FUNCTION SORT FOR RPG/3000      (XSORT)

(C) HEWLETT-PACKARD CO.  1980   WED, MAR 11, 1981, 9:16 AM

00000HSORTR      4A         X 33

00000NSORTR      4A         X 33

00005I C       33NECX
```

```
00005I C      33NECX

00010FNC    6    9

00010FNC    6    9

00020FDC    1   32

00020FDC    1   32

00030FDV          *

00030FDV          *

: EOD

RECORDS READ = 5

RECORDS SELECTED & SORTED = 4

  NUMBER OF COMPARES = 7

  SCRATCHFILE  I/O = 2

  CPU SECONDS = .440

  ELAPSED SECONDS = 1.884

TOTAL CPU SECONDS = 1.779

END OF PROGRAM

:SAVE TSTOUT
```

1 After the command to run XSORT is given, the program prints an identification banner. When it is completed, the user enters the specifications for the session.

2 The specifications, which the user enters without prompting from XSORT, are echoed back to the terminal. (Note that the echoing can be eliminated by specifying a Print Option of 1, 2, or 3 in Header column 27.)

3 The user must supply the colon before the EOD.

Example 4: Using XSORTEXT, Case 1

Single input to different output. Both XSORTIN and XSORTOUT are disk files.

```
:PURGE TSTOUT

:FILE XSORTIN=TSTIN

:FILE XSORTOUT=TSTOUT

:FILE XSORTEXT=TSTSPEC

:RUN XSORT.PUB.SYS

HP32104A.05.00    EXTRA  FUNCTION SORT FOR RPG/3000     (XSORT)

(C) HEWLETT-PACKARD CO.  1980   WED, MAR 11, 1981, 9:21 AM

00000HSORTR      4A        X  33

00005I C      33NECX

00010FNC    6    9                 Specifications

0002OFDC    1   32                   from TSTSPEC
```

```
0003OFDV         *

RECORDS READ = 5

RECORDS SELECTED & SORTED = 4

   NUMBER OF COMPARES = 7

   SCRATCHFILE  I/O = 2

   CPU SECONDS = .339

   ELAPSED SECONDS = 1.447

TOTAL CPU SECONDS = 1.116

END OF PROGRAM

:SAVE TSTOUT
```

Example 5: Using XSORTEXT, Case 2

Single input to different output. Input from $STDIN and output to $STDLIST.

The file equation--XSORTIN=$STDIN, in this example--means the program does not know in advance how many records are to be sorted since input is not from a disk file. As a result, it is necessary to enter an S Option line, specifying "Maximum number of records to be sorted" in columns 10-16.

```
:PURGE TSTOUT

:FILE XSORTIN=$STDIN

:FILE XSORTOUT=$STDLIST

.FILE XSORTEXT=TSTSPEX

:RUN XSORT.PUB.SYS

HP32104A.05.00    EXTRA FUNCTION SORT FOR RPG/3000     (XSORT)

(C) HEWLETT-PACKARD CO. 1980   WED, MAR 11, 1981, 9:24 AM

00000HSORTR     4A         X  33

00000S Y      35

00005I C      33NECX               Specifications

00010FNC   6  9                  from TSTSPEX

00020FDC   1 32

00030FDV        *

AB123GSD212345   JONES           X

AB234CICO12345   ALBERS            Interactive

AC102GSD100001   WOODSON           input from

AC321ABCD10001   FRANKEN           $STDIN

BB222CICO12345   WOODS

:EOD

AC321ABCD10001   FRANKEN        *
```

```
AB234CICO12345   ALBERS         *  Output to
BB222CICO12345   WOODS          *  $STDLIST
AC102GSD100001   WOODSON          *
RECORDS READ = 5
RECORDS SELECTED & SORTED = 4
    NUMBER OF COMPARES = 7
    SCRATCHFILE I/O = 2
    CPU SECONDS = .000
    ELAPSED SECONDS = 101.440
TOTAL CPU SECONDS = 1.220
END OF PROGRAM
```

Example 6: Job stream With Specifications Imbedded

Single file, in place sort.

```
!JOB TSTXSRT6,MGR.SUBSYS,XSORT;OUTCLASS=,1
!COMMENT   THIS IS A SINGLE FILE, INPLACE SORT
!COMMENT   THAT IS STREAMED. SPECS ARE IMBEDDED.
!COMMENT**************************************
!COMMENT**************************************
!FCOPY FROM=TSTIN;TO=
!FILE XSORTIN=TSTIN
!FILE XSORTOUT=TSTIN
!RUN XSORT.PUB.SYS
00000HSORTR     4A       X 33
00005I C      33NECX            Specifications
00010FNC   6   9              imbedded in
0002OFDC   1  32               job stream
0003OFDV         *              EDITOR file
!FCOPY FROM=TSTIN;TO=
!EOJ
```

1 No purge of the XSORTOUT file is required.

2 XSORTIN and XSORTOUT are both equated to TSTIN because this is an inplace sort.

Example 7: Job Stream With Specifications Imbedded.

Single file to different output that already exists.

```
!JOB TSTXSRT7,MGR.SUBSYS,XSORT;OUTCLASS=,1
```

```
!COMMENT THIS IS AN EXAMPLE OF A SINGLE SORT FILE TO
!COMMENT DIFFERENT OUTPUT THAT ALREADY EXISTS.
!COMMENT IT IS STREAMED, SPECS IMBEDDED
!COMMENT ***************************************
!COMMENT ***************************************
!FCOPY FROM=TSTIN;TO=
!FCOPY FROM=TSTFILE;TO=
!FILE XSORTIN=TSTIN
!FILE XSORTOUT=TSTFILE
!RUN XSORT.PUB.SYS
00000HSORTR     4A         X 33
00000S Y                    Specifications
00005I C      33NECX          imbedded in
00010FNC   6   9             job stream
00020FDC   1  32              EDITOR file
00030FDV        *
!FCOPY FROM=TSTFILE;TO=
!EOJ
```

1 No purge of the XSORTOUT file is required.

2 In this sort, the S Option must be employed to allow the program to overlay the existing file, TSTFILE.

NOTE After this job is run, the old contents of TSTFILE will have been erased, and replaced by the same output data produced by the other examples.

Example 8: Job Stream With Specifications Imbedded.

Multiple input file.

```
!JOB TSTXSRT8,MGR.SUBSYS,XSORT;OUTCLASS=,1
!COMMENT THIS IS A SORT USING MULTIPLE-
!COMMENT INPUT FILES. IT IS STREAMED,SPECS
!COMMENT IMBEDDED.************************
!COMMENT **********************************
!FCOPY FROM=TSTIN1;TO=
!FCOPY FROM=TSTIN2;TO=
!FCOPY FROM=TSTIN3;TO=
!PURGE TSTOUT
!FILE XSORTIN1=TSTIN1             Multiple input
```

```
!FILE XSORTIN2=TSTIN2            file designators

!FILE XSORTIN3=TSTIN3

!FILE XSORTOUT=TSTOUT

!RUN XSORT.PUB.SYS

00000HSORTR    4A       X 33

00000S Y     35                 Specifications

00005I C     33NECX              imbedded in

00010FNC   6   9                job stream

00020FDC   1  32                 EDITOR file

00030FDV        *

!EOD

!SAVE TSTOUT

!FCOPY FROM=TSTOUT;TO=

!EOJ
```

1 S Option is used to provide the maximum number of records to be sorted.

NOTE The example uses XSORTIN1, XSORTIN2, etc. In a multiple input sort, XSORTIN cannot be used to designate an input file. Input files must begin with XSORTIN1, but may skip numbers. For example, XSORTIN3 could follow XSORTIN1.

Example 9: SORTC (Count-Only Sort)

Single input file, no output.

```
!JOB TSTXSRT9,MGR.SUBSYS,XSORT;OUTCLASS=,1

!COMMENT THIS IS AN EXAMPLE OF A COUNT-ONLY SORT FROM

!COMMENT WHICH THERE IS NO OUTPUT OTHER THAN A STATISTICAL

!COMMENT REPORT ON THE RECORDS SELECTED TO SORT.*******

!COMMENT***********************************************

!FCOPY FROM=TSTIN;TO=

!FILE XSORTIN=TSTIN

!RUN XSORT.PUB.SYS     Specifications

00000HSORTC           imbedded in

000051 C     33NECX   job stream

!EOD                 EDITOR file

!EOJ
```

1 No purge of the XSORTOUT file is required.

2 No output file is designated since there is only a statistical report produced by a SORTC on the records selected for processing.

NOTE Count-only (SORTC) can also be used with multiple file sorts. See Example 8 for uses of multiple input files (XSORTIN1...XSORTING). You can combine the commands in Example 8 with those above to produce a Count-only, multiple file sort.

Here is a physically condensed copy of Example 9 after it was run. Note the statistical report on the records sorted.

```
:JOB TSTXSRT9,MGR.SUBSYS,XSORT

PRIORITY = DS; INPRI = 8; TIME = UNLIMITED SECONDS

JOB NUMBER = #J58

WED, MAR 11, 1981, 9:40 AM

HP3000 / MPE IV C.00.P8

 D U K E

:COMMENT THIS IS AN EXAMPLE OF A COUNT-ONLY SORT FROM

:COMMENT WHICH THERE IS NO OUTPUT OTHER THAN A STATISTICAL

:COMMENT REPORT ON THE RECORDS SELECTED TO SORT.*******

:COMMENT***********************************************

:FCOPY FROM=TSTIN;TO=

HEWLETT-PACKARD 32212A.3.13 FILE COPIER WED, MAR 11, 1981, 9:40 AM

(C) HEWLETT-PACKARD CO. 1980

*200*

WARNING: FROMFILE RECSIZE IS 72 BYTES, TOFILE RECSIZE IS 133 BYTES.

AB123GSD212345    JONES         X

AB234CICO12345    ALBERS

AC102GSD100001    WOODSON

AC321ABCD10001    FRANKEN

BB222CICO12345    WOODS

EOF FOUND IN FROMFILE AFTER RECORD 4

5 RECORDS PROCESSED *** 0 ERRORS

END OF SUBSYSTEM

:FILE XSORTIN=TSTIN

:RUN XSORT.PUB.SYS

HP32104A.05.0O    EXTRA FUNCTION SORT FOR RPG/3000    (XSORT)

(C) HEWLETT-PACKARD CO. 1980    WED, MAR 11, 1981, 9:40 AM

00000HSORTC

00005I C      33NECX

RECORDS READ = 5

RECORDS SELECTED = 4        Statistics
```

```
TOTAL CPU SECONDS = .224

END OF PROGRAM

:EOJ

CPU SEC. = 4. ELAPSED MIN. = 1. WED, MAR 11, 1981, 9:41 AM
```

8 Chapter 8 XSORT Applications

Introduction

There are three applications in this section. The first is a simple use of mixed include and omit sets in a single section. The second application involves multiple sections and sets. In both, the "comments" are used to make the sorts somewhat self-explanatory, and additional notes are provided to clarify what takes place.

Application #1

The file containing the specifications for the sort is designated TSTSPECA.

```
     HSORTR      5D              OX     72
     I C        3EQC1 } include set   \             INCLUDE IF 1 IN POSITION 3
     IAC        5EQC1 }               | one section      AND 5
     O C        3EQC1 } omit set      |             OMIT OTHERS WITH 1 IN  POSITION
     I C        7NEC1 } include set   /             INCLUDE IF NOT 1 IN POSITION  7
     FNC    9   13
     FNC    9   13
     FDC    1   72
```

The comment section for the file explains what will be included and omitted by the sort. Note that the two include sets and one omit set form a single Record Section. Also, the sorted file will be in descending order.

The input file, designated TSTINA, has comments which tell you whether each record will be included or omitted, according to the specifications in TSTSPECA.

```
     Positions            3 5 7 9-13      18-72

                          1 1 1 A0001     =INCLUDE (1 IN POS.3 AND POS.5)
                          1 1 0 H9999     =INCLUDE (1 IN POS.3 AND POS.5)
                          1 0 1 N1045     =OMIT (1 IN POS.3)
                          1 0 0 B0002     =OMIT (1 IN POS.3)
                          0 1 1 A7523     =NOT INCLUDED (1 IN POS.7)
                          0 1 0 R3316     =INCLUDE (NOT 1 IN POS.7)
                          0 0 1 T9458     =NOT INCLUDED (1 IN POS.7)
                          0 0 0 B3257     =INCLUDE (NOT 1 IN POS.7)
```

Positions 3, 5, and 7 of the records contain different arrangements of 1's and 0's, and positions 9-13 are filled with information which will determine the location of the sorted records with regard to the descending order.

Here is the output file for the sort with the records marked to be omitted missing.

```
                    0 1 0 R3316          =INCLUDE  (NOT 1 IN POS.7)
                    1 1 0 H9999          =INCLUDE  (1 IN POS.3 AND POS.5)
```

```
0 0 0 B3257          =INCLUDE  (NOT 1 IN POS.7)
1 1 1 A0001          =INCLUDE  (1 IN POS.3 AND POS.5)
```

The descending order is evident in the characters of the control field described in positions 9-13.

Application #2

This application shows how multiple sections and sets may be applied in a sort. Again, the comments are used to earmark the records and the specifications for the results of the sort.

To help you understand the way Record Types, Sections, and Sets are related refer to Figure 8-1.

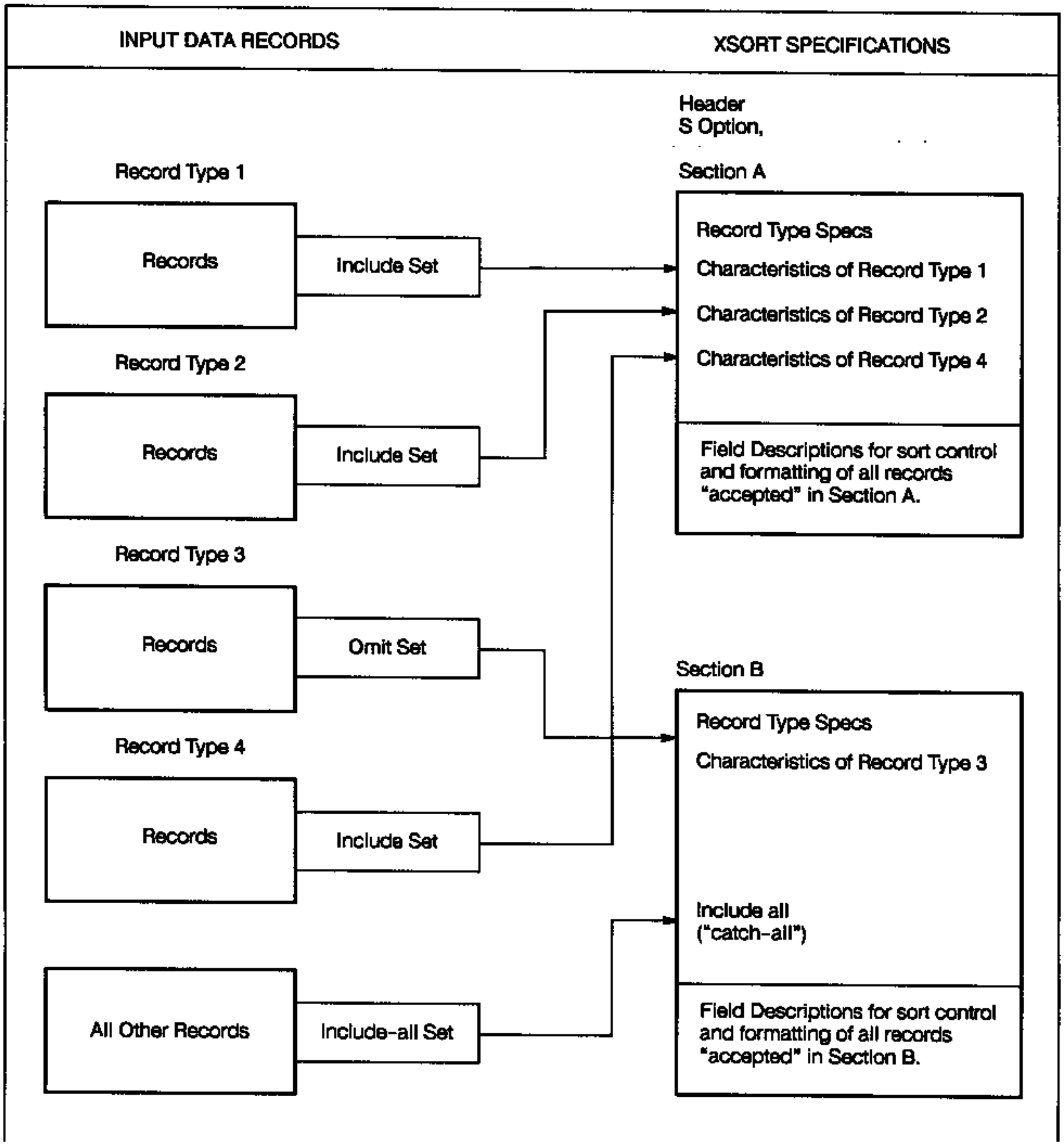

INPUT DATA RECORDS
XSORT SPECIFICATIONS
Header
S Option,
Record Type 1
Records
Include Set
Record Type 2
Records
Include Set
Record Type 3
Records
Omit Set
Record Type 4
Records
Include Set
All Other Records
Include-all Set
Section A
Record Type Specs
Characteristics of Record Type 1
Characteristics of Record Type 2
Characteristics of Record Type 4
Field Descriptions for sort control and formatting of all records "accepted" in Section A.
Section B
Record Type Specs
Characteristics of Record Type 3
Include all ("catch-all")
Field Descriptions for sort control and formatting of all records "accepted" in Section B.

Figure 8-1. Relating Record Types, Sections, and Sets

The specifications for the sort are supplied by the file designated TSTSPECB.

```
     HSORTR    13A         OX    80       EXAMPLE USING MIXED SECTIONS & SETS.
     S Y      75                            OVERLAY EXISTING OUTPUT; 75 RECS MAX
     I*
     I C       6EQCX                    SECTION-A, TYPE 1
     IAC  23  24LEC50                     (INCLUDE X'S BELOW 50)
     IOC       6EQCY                    SECTION-A, TYPE 2
     IAC  23  24LEC50                     (INCLUDE Y'S BELOW 50)
     IOC       6EQCM                    SECTION-A, TYPE 4
     IAC  23  24LEC50                     (INCLUDE M'S BELOW 50,
     IAC  55  58GTF  51  54                  AND ABOVE LIMIT)
     F*                                 SECTION-A FIELD DESCRIPTIONS:
     FNC   2   5                          SORT BY ITEM#, AND
     FNC       1                            CLASS
     FDV                                  OUTPUT BLANK IN POSITION 1
     FDC   1  71                          OUTPUT DATA (FROM INPUT 1-71)
     O*
     O C       6EQCZ                    SECTION-B, TYPE 3
     OAC  12  13NECCA                     (OMIT Z'S NOT IN CALIFORNIA)
     I                                  SECTION-B, INCLUDE ALL OTHER RECORDS
     F*                                 SECTION-B FIELD DESCRIPTIONS:
     FNC   2   5                          SORT BY ITEM#,
     FNC       1                            CLASS, AND
     FNC  73  80                            LINE#
     FDV         *                        OUTPUT ASTERISK IN POSITION 1
     FDC   1  71                          OUTPUT DATA (FROM INPUT 1-71)
     FDC  73  80                          OUTPUT ORIGINAL LINE#'S
```

Because this is a multiple file sort, the S Option is used. The Y entered in column 8 indicates that you are willing to have the output file overlay an already existing file.

There are two sections, A and B, set off by the specifications. Section A is comprised of three Include Sets, plus control fields and data fields. A blank forced in position 1 of the output will "flag" Section A.

Section B is comprised of an Omit Set, an Include-all Set, and the control and data fields. An asterisk forced into position 1 of the output record "flags" Section B.

This is the input file, designated TSTINB, for the sort.

```
        Item               #Size Code
         |                     |
    Class|Group  State         |    Description     Limit Cost              Line #
      |  | |      |            |          |              |                       |
      /\/-\/\     /\           /\ /---------------\ /-------\               /------\
      A0001X      CA           01 SECTION-A, TYPE-1  05001021  00001000
      A0113Z      TX           21 SECTION-B, TYPE-3  09000899  00002000
      A9059Z      CA           57 SECTION-B, OTHER   05002001  00003000
```

```
   B0001X      TX          60 SECTION-B, OTHER   09000112  00004000
   B0113M      FL          20 SECTION-A, TYPE-4  08000860  00005000
   B9001M      CA          51 SECTION-B, OTHER   05000525  00006000
   B9001M      CA          72 SECTION-B, OTHER             05000501       00007000
   09059Z      FL          01 SECTION-B, TYPE-3            08000801       00008000
   C0001Y      CA          99 SECTION-B, OTHER             05000602       00009000
   C0113Z      NY          20 SECTION-B, TYPE-3            02503000       00010000
   C9001M      CA          99 SECTION-B, OTHER             05009999       00011000
   C9001Y      OR          17 SECTION-A, TYPE-2            06000250       00012000
   C9059Z      CA          10 SECTION-B, OTHER             05000980       00013000
   C9059z      CA          35 SECTION-B, OTHER             05000500       00014000
   C9059Z      CA          87 SECTION-B, OTHER             05000970       00015000
   D0001Y      FL          02 SECTION-A, TYPE-3            08000700       00016000
   D0113M      TX          09 SECTION-B, OTHER             09000850       00017000
   D9059Y      NY          50 SECTION-A, TYPE-2            02502140       00018000
   W9001X      FL          13 SECTION-A, TYPE-1            08000750       00019000
  1-6           12            23  26                            51      58          73     80
     Positions
```

The record portion designated "description" is used to highlight the purpose of the sort. Both sections will be sorted by item and class, and in Section B line numbers will also be used.

Here is the output file produced by the sort.

```
     A0001X     CA      01 SECTION-A, TYPE-1        05001021
   * B0001X     TX      60 SECTION-B, OTHER         09000112             00004000
   * C0001Y     CA      99 SECTION-B, OTHER         05000602             00009000
     D0001Y     FL      02 SECTION-A, TYPE-3        08000700
     B0113M     FL      20 SECTION-A, TYPE-4        08000860
   * D0113M     TX      09 SECTION-B, OTHER         09000850             00017000
   * B9001M     CA      51 SECTION-B, OTHER         05000525             00006000
   * B9001M     CA      72 SECTION-B, OTHER         05000501             00007000
     C9001Y     OR      17 SECTION-A, TYPE-2        06000250
   * C9001M     CA      99 SECTION-B, OTHER         05009999             00011000
     W9001X     FL      13 SECTION-A, TYPE-1        08000750
   * A9059Z     CA      57 SECTION-B, OTHER         05002001             00003000
   * C9059Z     CA      10 SECTION-B, OTHER         05000980             00013000
   * C9059Z     CA      35 SECTION-B, OTHER         05000500             00014000
   * C9059Z     CA      87 SECTION-B, OTHER         05000970             00015000
     D9059Y     NY      50 SECTION-A, TYPE-2        02502140
```

NOTE The file is in order first by item# (output positions 3-6), then by class number (output position 2), and then by line# (positions 73-80) for records in Section B. In addition, Section B records are "flagged" by an asterisk in output position 1 and by the input line# in position 73-80.

Application #3

This application is in three parts. Each part is a sample job typical of the work XSORT was designed to address in the field. The jobs use the same files in a variety of ways for different sort requirements. Each of the three sample jobs included is presented with an introduction, XSORT specifications entered on the form, and a discussion of the specifications.

A breakdown of the files to be sorted and their contents precedes the discussion.

Files	Records in File	Fields Used in Sorts
Weekly Sales	Detail sales records.	Date, customer, product, quantity
Weekly Bookings	Detail booking records.	Date, customer, product, quantity
Sales and Bookings	Summary sales and history	Date, customer, product, bookings. quantity.

Record Format for Weekly Sales File

Field Names	Contents	Record Positions
RECCDE	Record "WS"	1 -2
INVNUM	Invoice Number	3 -8
INVDAY	Invoice data - day	9 -10
INVMON	Invoice data - month	11 -12
INVYR	Invoice data - year	13 -14
CUSTNO	Customer number	15 -24
CUSTTY	Customer type	25 -25
MRCHLN	Merchandise line	26 -27
PRODCO	Product code	28 -37
UNITS5	Sales units	38 -42
STACDE	State code	43 -44
SLSTER	Sales territory	45 -47
SLSNMl	Sales number	48 -49
TRANCD	Sales type	50 -50

Record Format for Weekly Bookings File

Field Names	Contents	Record Positions
RECCDE	Record code "WB"	1 -2
CUSTNO	Customer numbe	3 -12
CUSTTY	Customer type	13 -13
ORDRNO	Order number	14 -19
ORTYCD	Order type	20 -20

Field Names	Contents	Record Positions
MRCHLN	Merchandise line	21 -22
PRODCO	Product code	23 -32
STACDE	State code	33 -34
SLSTER	Sales territory	35 -36
SLSNM1	Salesman number	37 -39
CANIND	Cancel indicator	40 -40
DATEMM	Date - month	41 -42
DATEYY	Date - year	43 -44
UNITS5	Units booked	45 -49
TRANCD	Transaction code	50 -50

Record Format for the Sales History File

Field Names	Contents	Record Positions
RECCDE	Record code "SB"	1 -2
CUSTNO	Customer number	3 -12
STACDE	State code	13 -14
SLSTER	Sales territory	15 -16
SLSNM1	Salesman number	17 -19
PRODCD	Product code	20 -29
CANIND	Cancellation indicator	30 -30
GARTCD	Garment type code	31 -31
DATEMM	Date - month	32 -33
DATEYY	Date - year	34 -35
UNITSB	Units booked	36 -42
UNITSHP	Units shipped	43 -49
UNTCAN	Units cancelled	50 -56
UNTRET	Units returned	57 -63
MRCHLN	Merchandise line	64 -65
DOLBOK	Booked value	66 -74

Field Names	Contents	Record Positions
DOLSHP	Shipped value	75 -83
DOLCAN	Cancelled value	84 -92
DOLRET	Returned value	93 -101

These are the files that were created specifically for the sample sorts. They are the Weekly Sales file (WKSALE), the Weekly Bookings file (WKBOOK), and the Sales History file (HISTUPD), in that order.

```
WB111111111119999999105ZZZ9999999CA0110110181123459

WB211111111128999998205XXX8888888TX0120210281234568

WB311111111137999997305LLL7777777AZ0130311280345677

WB411111111146999996405VVV6666666NV0140411180456786

WB511111111155999995505AAA5555555CA0150510181567895

WB611111111164999994605DDD4444444CA0160610281678904

WB711111111173999993709JJJ3333333CA81707N0281789013

WB811111111182999992809GGG2222222CA91808N0181890122

WB911111111191999991909ZZZ1111111CA11909I0181901231

WB011111111100999990009VVV0000000CA21101N0281123450

WB111111111119999999109ZZZ9999999CA31202N0281234561

WB211111111128999988209XXX8888888CA41303N1280345672
WS9000123102811111111111905ZZZ999999912345CA011019

WS90001130028121111111112805XXX888888823456CA011028

WS9000102901813111111111113705LLL777777734567TX011037

WS900009280181411111111114605VVV666666645678AZ011046

WS900008270181511111111115505AAA555555556789CA011055

WS900007260181611111111116405DDD444444467890CA011064

WS900006250181511111111115309JJJ333333378901CA031073

WS900005240181811111111118209GGG222222289012CA031082

WS900004230181911111111119109ZZZ111111190123CA031091

WS900003220181011111111110009VVV000000001234CA031100

WS90000221018112111111112109ZZZ999999923456CA031111
```

```
WS900001200181231111111113209XXX888888834567CA031122

SB1111111111CA11101ZZZ99999991102811234567234567834567891234567051234567890

SB2111111112CA01202XXX88888881202812345678345678945678905678901092345678900

SB3111111113TX01303LLL77777771302813456789456789056789016789012093456789010

SB4111111114CA01404VVV66666661401814444444444444464444444444447094444444440

SB5111111115CA01505AAA55555551502815555551555555255555535555554055555555510

SB6111111116CA01606DDD44444441602816666661666666266666636666664056666666610

SB7111111117CA01707JJJ33333339702817777771777777277777737777774097777777710

SB8111111118CA01808GGG22222229802818888881888888288888838888884058888888810

SB9111111119CA02909ZZZ11111112901819999991999999299999939999994099999999910

SB0111111110CA04101VVV00000001002810000001000000200000030000004050000000010

SB1111111111CA01202ZZZ99999997102811111111111111211111131111114061111111110

SB2111111112AZ01303XXX88888881202812222221222222222222223222222409222222210
```

Sample Sort 1

This sort is used to select cancelled order records from the Weekly Bookings file. It produces a Weekly Cancellation Analysis. The "cancellation indicator" (in position 40), which is equal to 1, provides selection criteria.

Output records will contain only those fields required for the report. The output file will be sequenced by Customer Number (CUSTNO) and subsequently by Product Code (PRODCO).

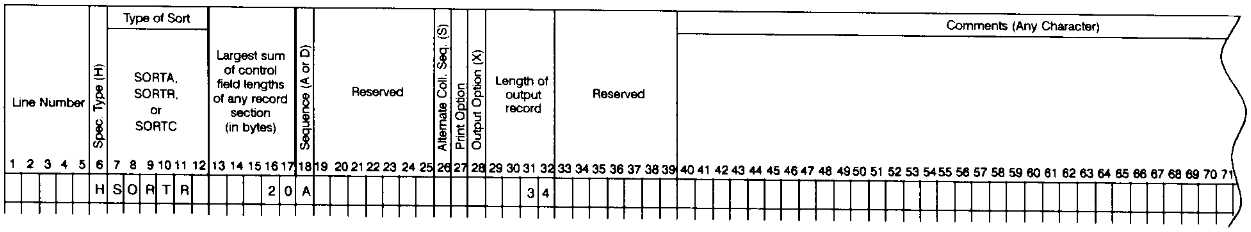
Here are the specifications as they would appear at the terminal and on the specification form.

```
HSORTR        20A           34

I C          40EQC1                      SELECT CAN.ORD.

I*                                    IND.= 1

FNC    3     12                    CUSTNO  CUSTOMER #

FNC   23     32                    PRODCD  PRODUCT CODE

FDC   13     13                    CUSTTY  CUSTOMER TYPE

FDC   20     20                    ORDTYP  ORDER TYPE

FDC   37     39                    SLSNM1  SALESMAN NUMBER

FDC   41     42                    DATEMM  DATE MONTH

FDC   43     44                    DATEYY  DATE YEAR

FDC   45     49                    UNITS5  ORDER UNITS
```

HEWLETT PACKARD

XSORT SPECIFICATIONS

Header

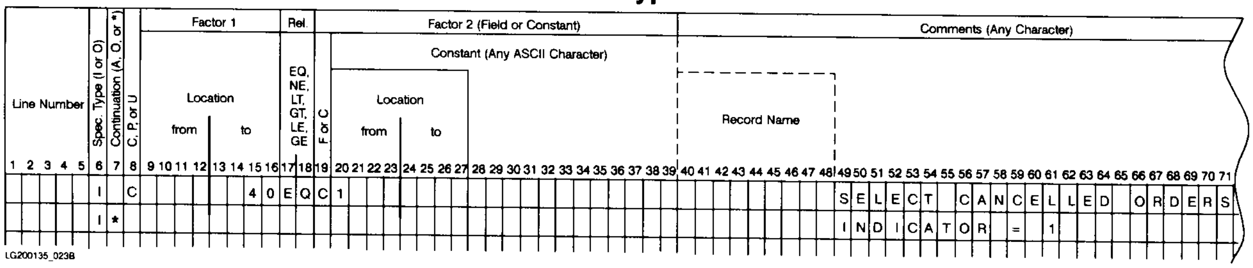


Line Number (1-5)	Spec. Type (H) (6)	Type of Sort: SORTA, SORTR, or SORTC (7-12)	Largest sum of control field lengths of any record section (in bytes) (13-17)	Sequence (A or D) (18)	Reserved (19-25)	Alternate Coll. Seq. (S) (26)	Print Option (27)	Output Option (X) (28)	Length of output record (29-32)	Reserved (33-39)	Comments (Any Character) (40-71)
	H	SORTR	20	A					34		

← Insert ALTSEQ specifications here (if used).
← Insert S OPTION specification here (if used).

LG200135_023A

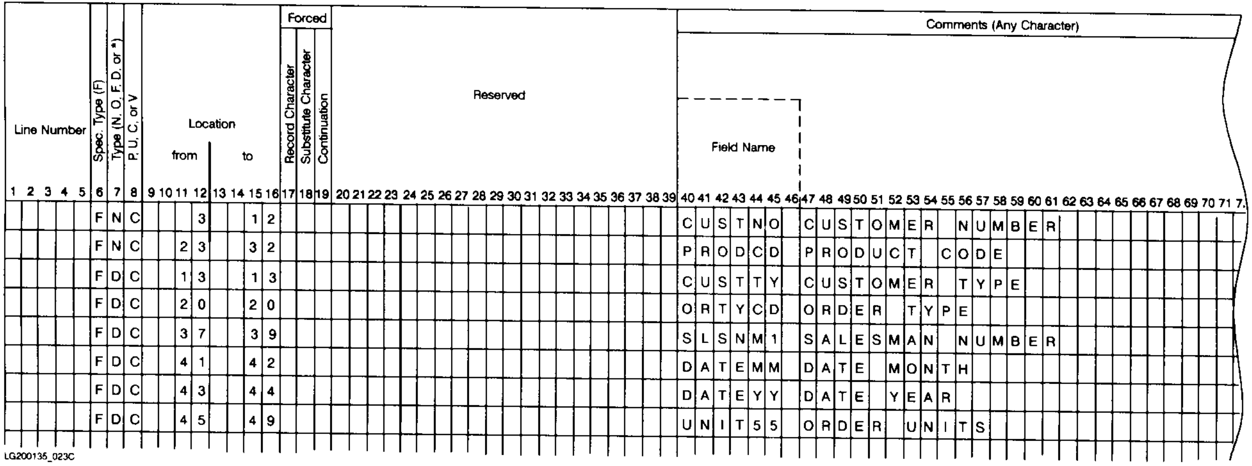
Record Type



Line Number (1-5)	Spec. Type (I or O) (6)	Continuation (A, O, or *) (7)	C, P, or U (8)	Factor 1 Location from (9-12)	Factor 1 Location to (13-16)	Rel. EQ, NE, LT, GT, LE, GE (17-18)	F or C (19)	Factor 2 (Field or Constant): Constant (Any ASCII Character) Location from (20-23)	Location to (24-27)	(28-39)	Comments (Any Character): Record Name (40-71)
	I	C			40	EQ	C	1			SELECT CANCELLED ORDERS
	I	*									INDICATOR = 1

LG200135_023B

Field Description



Line Number (1-5)	Spec. Type (F) (6)	Type (N, O, F, D, or *) (7)	P, U, C, or V (8)	Location from (9-12)	Location to (13-16)	Forced: Record Character (17)	Substitute Character (18)	Continuation (19)	Reserved (20-39)	Comments (Any Character): Field Name (40-71)
	F	N	C	3	12					CUSTNO CUSTOMER NUMBER
	F	N	C	23	32					PRODCD PRODUCT CODE
	F	D	C	13	13					CUSTTY CUSTOMER TYPE
	F	D	C	20	20					ORTYCD ORDER TYPE
	F	D	C	37	39					SLSNM1 SALESMAN NUMBER
	F	D	C	41	42					DATEMM DATE MONTH
	F	D	C	43	44					DATEYY DATE YEAR
	F	D	C	45	49					UNIT55 ORDER UNITS

LG200135_023C

Explanation of Specifications.

Header Specification.

Col 6 The H entry identifies this as the Header line.

Cols 7-12 SORTR identifies this as a Record-out sort.

Cols 13-17 Total length of the sort control fields for this sort is 20. The total is derived from adding CUSTNO (10 characters) and PRODCO (10 characters).

Col 18 The A means the output sequence will be in ascending order by CUSTNO and PRODCO.

Cols 29-32 34 is the total length of output records. The control fields are not being dropped in this sort so their total (20 characters) is added to the 14 character sum derived from adding the six data fields in the sort.

Record Type Specifications.

Column 6 The I identifies this an an include line. Input records which match the criteria specified by this line will be used in the sort. Those which fail to meet the criteria will be omitted.

Column 7 The * indicates a comment line.

Column 8 Use only complete characters--one byte (eight bits)--in making comparisons.

Cols 9-12 Because the input record field for factor 1 is only one character long, this entry is left blank.

Cols 13-16 Input record position where factor 1 field begins and ends is position 40.

Cols 17-18 EQ means factor 1 must equal factor 2.

Column 19 The C means factor 2 is a constant.

Column 20 This is where the value of the constant is entered. In this instance, 1 is the factor 2 constant.

Field Description Specifications.

Column 6 The F identifies this as a field specification line.

Column 7 The N indicates that this is a normal control field. The D's in the column define the data fields.

Column 8 The C tells the sort to use complete characters--1 byte (8 bits).

Cols 9-12 3, 23, and the other numbers indicate record positions where field start. They are the "from" positions.

Cols 13-16 These are the positions in the records where the fields end. They are the "to" positions.

Here is the output of the sort.

```
1111111111AAA5555555555505018156789

2111111112DDD4444444446606028167890

2111111112XXX8888888882202028123456

4111111114VVV6666666664404118123457

4111111114ZZZ9999999991101018112345

7111111117LLL7777777773303128123456
```

Sample Sort 2

This sort produces a record address file for the selected records of the Sales History File. Records are selected when they contain specific combinations of Sales Territory (SLSTER) and Merchandise Line (MRCHLN). The output of this sort is a record address file containing relative record numbers for the selected records. The sequences will be Garment Type Code and Merchandise Line within the garment type. A hex dump of the output is followed by a simple RPG program which utilizes the output in a meaningful report.

Two additional tasks are performed here:

1 It is desired to combine the Garment Type Codes 2 and 3 into a single category for sorting purposes. Therefore, a "3" will be substituted for a "2" wherever it is encountered in the Garment Type code field. (This is a conditional force on the control field.)

2. An opposite sort is performed on units booked (UNITSB) to put the bookings in descending sequence within Merchandise Line.

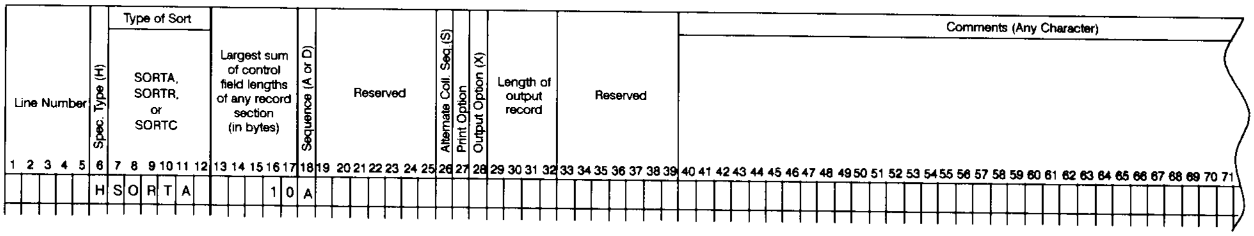
Here are the specifications as they would appear at the terminal and on the specification form.

```
HSORTA       10A

I C  15      16EQC01                        SELECT TERR. 01

IAC  64      65EQC05                        MERCH. LINE 05

IOC  15      16EQC01                        OR

IAC  64      65EQC09                        SELECT TERR. 01

I*                                   MERCH. LINE 09

FNC          31                         FORCE TO EQ.CD.2

FFC          3123X               GARTCD      AND CODE 3

FNC  64      65                  MRCHLN      MERCHANDISE LINE

FOC  36      42                  UNITSB      UNITS BOOKED
```

HEWLETT PACKARD

XSORT SPECIFICATIONS

Header



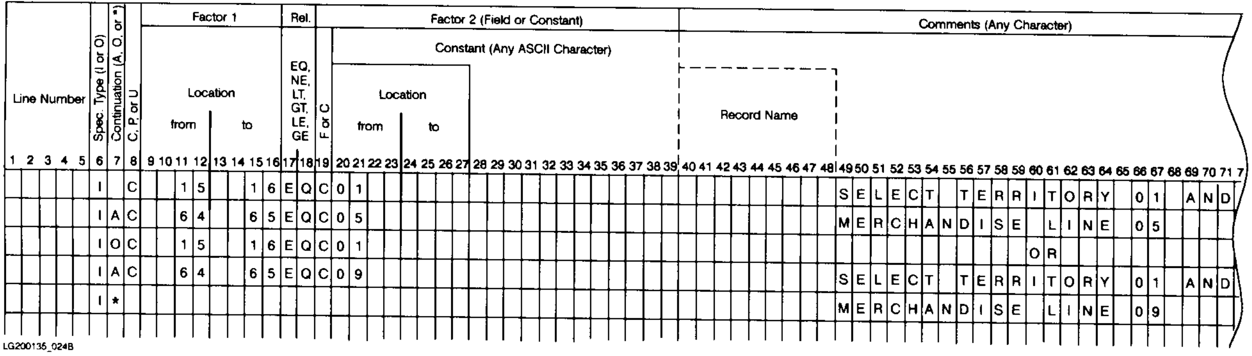
Line Number	Spec. Type (H)	Type of Sort: SORTA, SORTR, or SORTC	Largest sum of control field lengths of any record section (in bytes)	Sequence (A or D)	Reserved	Alternate Coll. Seq.(S)	Print Option	Output Option (X)	Length of output record	Reserved	Comments (Any Character)
1 2 3 4 5	6	7 8 9 10 11 12	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 68 69 70 71
	H	SORTA	10	A							

← Insert ALTSEQ specifications here (if used).

← Insert S OPTION specifications here (if used).

LG200135_024A

Record Type

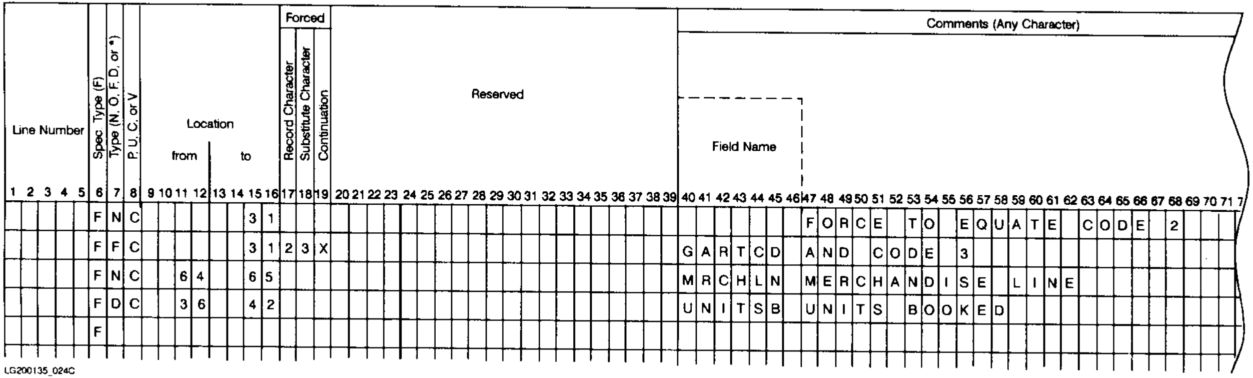


Line Number	Spec. Type (I or O)	Continuation (A, O, or *)	C, P, or U	Factor 1 Location from	Factor 1 Location to	Rel. EQ, NE, LT, GT, LE, GE	F or C	Factor 2 (Field or Constant) Location from	Factor 2 (Field or Constant) Location to	Comments (Any Character)
1 2 3 4 5	6	7	8	9 10 11 12	13 14 15 16	17 18	19	20 21 22 23	24 25 26 27	40 ... 71
	I		C	15	16	EQ	C	01		SELECT TERRITORY 01 AND
	I	A	C	64	65	EQ	C	05		MERCHANDISE LINE 05
	I	O	C	15	16	EQ	C	01		OR
	I	A	C	64	65	EQ	C	09		SELECT TERRITORY 01 AND
	I	*								MERCHANDISE LINE 09

Constant (Any ASCII Character) — Record Name

LG200135_024B

Field Description



Line Number	Spec. Type (F)	Type (N, O, F, D, or *)	P, U, C, or V	Location from	Location to	Forced Record Character	Forced Substitute Character	Continuation	Reserved	Field Name	Comments (Any Character)
1 2 3 4 5	6	7	8	9 10 11 12	13 14 15 16	17	18	19	20 ... 39	40 ... 46	47 ... 71
	F	N	C		31						FORCE TO EQUATE CODE 2
	F	F	C		31	2	3	X		GARTCD	AND CODE 3
	F	N	C	64	65					MRCHLN	MERCHANDISE LINE
	F	D	C	36	42					UNITSB	UNITS BOOKED
	F										

LG200135_024C

Explanation of Specifications.

Header Specification.

Column 6 The H identifies this as the Header line.

Cols 7-12 SORTA identifies this as an Address-out sort.

Cols 13-17 10 is the largest sum of all control field lengths. It is derived generally from adding all control fields designated with an F, N, or 0 in column 7 for each record type. This case is an exception because a forced control field is being used. It will overlay the position in the work record occupied by the character it is forcing and, therefore, will not contribute to the total control field length.

Column 18 The A indicates that the sorted records will be in ascending order by control fields.

Record Type Specifications.

Column 6 The I identifies this as an include line. The input records matching the criteria specified by this line will be used by the sort.

Column 7 Leaving this column blank indicates that this is the first line of an I (include) set of record type lines. The * means a comment line. A identifies an AND line. It signifies that these specifications continue the definitions of the record type described in the last record type entry. O entered here stands for OR. It means that these specifications define a different type of record than the one on the previous line.

Column 8 A "C" here means a character is used in comparisons.

Cols 9-12 The numbers 15 and 64, entered twice here, indicate the position where the factor 1 field begins.

Cols 13-16 The numbers 16 and 65 entered here indicate where the factor 1 field ends.

Cols 17-18 Factor 1 will be compared with factor 2, which is a constant in all cases here. EQ means you want the two factors to be equal.

Column 19 The C identifies factor 2 as a constant.

Cols 20-23 The entries 01, 05, 01, and 09 are two-character constants. The zeros are entered here because the control fields are all two characters long, and constants must be the same size as control fields. (You can leave the positions presently occupied by zeros blank. It is important, however, to keep the entries right justified.)

Field Description Specifications.

Column 6 The F identifies this as a field description line.

Column 7 The N identifies it as a normal control field. O means opposite control field. F indicates that one control field will be forced into position. In this case it will overlay the position in the work record occupied by the previous control field--if the conditions of the force operations are met.

Cols 9-12 The numbers entered in this column identify the starting position of a control field in a record.

Cols 13-16 These numbers identify the end position of a control field in a record. Note that 31, which is a field that is only one position in length, is only entered once, at the end position.

Column 17 The 2 is the character that must be in this control field in order for a 3 (defined in column 18) to be forced into the work record.

Column 18 The 3 is the character which will replace the control field in the work record if the control field equals 2 (defined in column 17).

Since this is an Address-out sort, a hex dump of the output is provided.

```
SRTUT RECORD 0 (%0, #0)

00000: 0000 0002

SRTUT RECORD 1 (%1, #1)

00000: 0000 0001

SRTUT RECORD 2 (%2, #2)

00000: 0000 000B
```

```
SRTUT RECORD 3 (%3, #3)

00000: 0000 0003
SRTUT RECORD 4 (%4, #4)

00000: 0000 0004
SRTUT RECORD 5 (%5, #5)

00000: 0000 0005
SRTUT RECORD 6 (%6, #6)

00000: 0000 0006

SRTUT RECORD 7 (%7, #7)

00000: 0000 0007
```

A simple RPG program has been devised to make use of the output of the sort.

```
PAGE 0001    HEWLETT PACKARD 32104A.04.07 RPG/3000 (C) HEWLETT-PACKARD CO. 1978.

 WED,  MAR  11, 1981, 10:24 AM

0001          $CONTROL   USLINIT

0002          H                                      1

0003          FHISTORY   IP  F   101R  I        DISK

0004          FRAF       IRE F     4  4 T      EDISK

0005          FREPORT    0   F   132     0V    LLP

0006          E    RAF       HISTORY

0007          LREPORT     66FL  60OL

0008          IHISTORY   NS  01

0009          I                                  3  12 CUSTNO

0010          I                                 31  31 GARTCD

0011          I                                 32  350DATE

0012          I                                 36  420UNITSB

0013          C            TOTAL    ADD UNITSB   TOTAL   90

0014          0REPORT     H  201    1P

0015          0          OR        0V

0016          0                       UDATE Y   16

0017          0                               51 "SALES HISTORY LISTING"

0018          0                       PAGE  Z   72
```

```
0019     0                   70  "PAGE"
0020     0      H  1     1P
0021     0     OR        0V
0022     0                   37  "GRMNT    UNITS"
0023     0                   58  "CUSTOMER     DATE"
0024     0      H  1     1P
0025     0     OR        0V
0026     0                   37  "-----    -----"
0027     0                   58  "----------   -----"
0028     0      D  1     01
0029     0              GARTCD    25
0030     0              UNITSBJ   38
0031     0              CUSTNO    50
0032     0              DATE  Y   58
0033     0      T  1     LR
0034     0              TOTAL J   38
```

Here is the output of the program. You will notice that the sequence under the heading "GRMNT" (GARMENT) seems to start in descending rather than ascending order. In fact, a 3 was substituted for each 2 when encountered in the Garment Type code field, and sorting was done according to these values.

```
3/11/81          SALES HISTORY LISTING          PAGE 1

        GRMNT     UNITS    CUSTOMER      DATE

        -----     -----   ----------     -----

         3     3,456,789   3111111113     2/81

         2     2,345,678   2111111112     2/81

         2     2,222,221   2111111112     2/81

         4     4,444,444   4111111114     1/81

         5     5,555,551   5111111115     2/81

         6     6,666,661   6111111116     2/81

         7     7,777,771   7111111117     2/81

         8     8,888,881   8111111118     2/81
              41,357,996
```

Sample Sort 3

This sample sort utilizes the multiple input file feature of XSORT. It combines the Weekly Bookings file and the Weekly Sales file. The sequence of the combined output file will be by customer number. Additionally, the Weekly Sales file is reformatted on output to the sorted file in order to match the format of the Weekly Bookings file records.

The file statements required for this sort are:

```
FILE XSORTIN1=WKBOOK

FILE XSORTIN2=WKSALE

FILE XSORTIN3=HISTUPD
```

Here are the specifications as they would appear at the terminal and on the specification form.

```
HSORTR     10A      X  90

S         36

I C    1    2EQCWB                      SEL.WKLY.BK.REC.

FNC    3    12                CUSTNO    CUSTOMER NUMBER

FDC    1    50

I C    1    2EQCWS                      SELECT WEEKLY

I*                                SALES RECORD

FNC   15    24                CUSTNO    CUSTOMER NUMBER

FDC    1    2                 RECCDE    RECORD CODE

FDC   15    24                CUSTNO    CUSTOMER NUMBER

FDC   25    25                CUSTTY    CUSTOMER TYPE

FDC    3    8                 INVNUM    INVOICE NUMBER

FDC   50    50                TRANCD    SALES TYPE

FDC   26    27                MRCHLN    MERCHANDISE LINE

FDC   28    37                PRODCD    PRODUCT CODE

FDC   43    44                STACDE    STATE CODE

FDC   45    47                SLSTER    SALES TERRITORY

FDC   48    49                SLSMN1    SALESMAN NUMBER

FDC   50    50                TRANCD    SALES TYPE

FDC   11    12                INVMON    INVOICEDATEMONTH

FDC   13    14                INVYR     INVOICEDATEYEAR

FDC   38    42                UNITSF    SALES UNITS
```

XSORT SPECIFICATIONS

Header

Line Number (1-5)	Spec. Type (H) (6)	Type of Sort: SORTA, SORTR, or SORTC (7-12)	Largest sum of control field lengths of any record section (in bytes) (13-17)	Sequence (A or D) (18)	Reserved (19-25)	Alternate Coll. Seq. (S) (26)	Print Option (27)	Output Option (X) (28)	Length of output record (29-32)	Reserved (33-39)	Comments (Any Character) (40-71)
	H	SORTR	10	A				X	90		

← Insert ALTSEQ specifications here (if used).
← Insert S OPTION specification here (if used).

LG200135_025A

S Option

Line Number (1-5)	Spec. Type (S) (6)	0 or 1 (7)	N or Y (8)	Reserved (9)	Maximum number of records to be sorted (10-16)	Reserved (17-39)	Comments (Any Character) (40-71)
					36		

LG200135_025D

Record Type

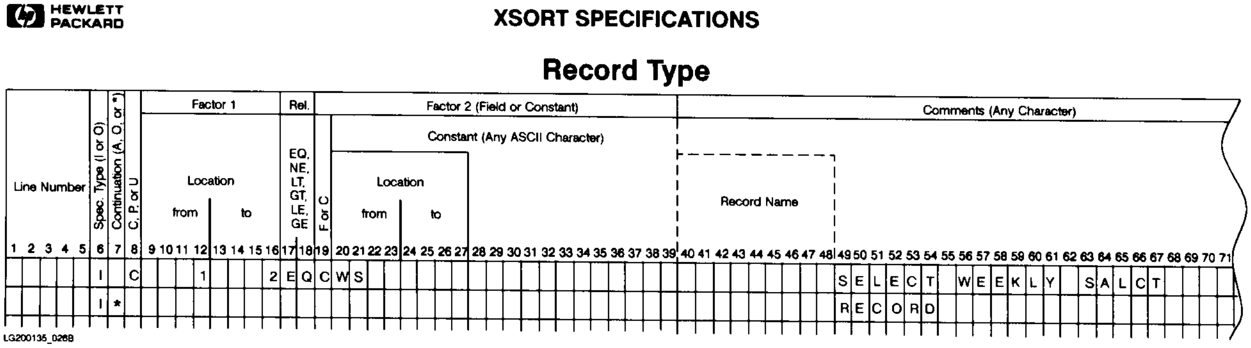
Line Number (1-5)	Spec. Type (I or O) (6)	Continuation (A, O, or *) (7)	C, P, or U (8)	Factor 1 Location from (9-12)	Factor 1 Location to (13-16)	Rel. EQ, NE, LT, GT, LE, GE (17-18)	F or C (19)	Factor 2 (Field or Constant): Location from (20-23) / Constant (Any ASCII Character) (20-39)	Factor 2 Location to (24-27)	Comments (Any Character): Record Name (40-71)
	I		C	1	2	EQ	C	WB		SELECT WEEKLY BOOKING RECORDS

LG200135_025B

Field Description

Line Number (1-5)	Spec. Type (F) (6)	Type (N, O, F, D, or *) (7)	P, U, C, or V (8)	Location from (9-12)	Location to (13-16)	Forced: Record Character (17)	Forced: Substitute Character (18)	Continuation (19)	Reserved (20-39)	Comments (Any Character): Field Name (40-70)
	F	N	C	3	12					CUSTNO CUSTOMER NUMBER
	F	D	C	1	50					
	F									

LG200135_025C



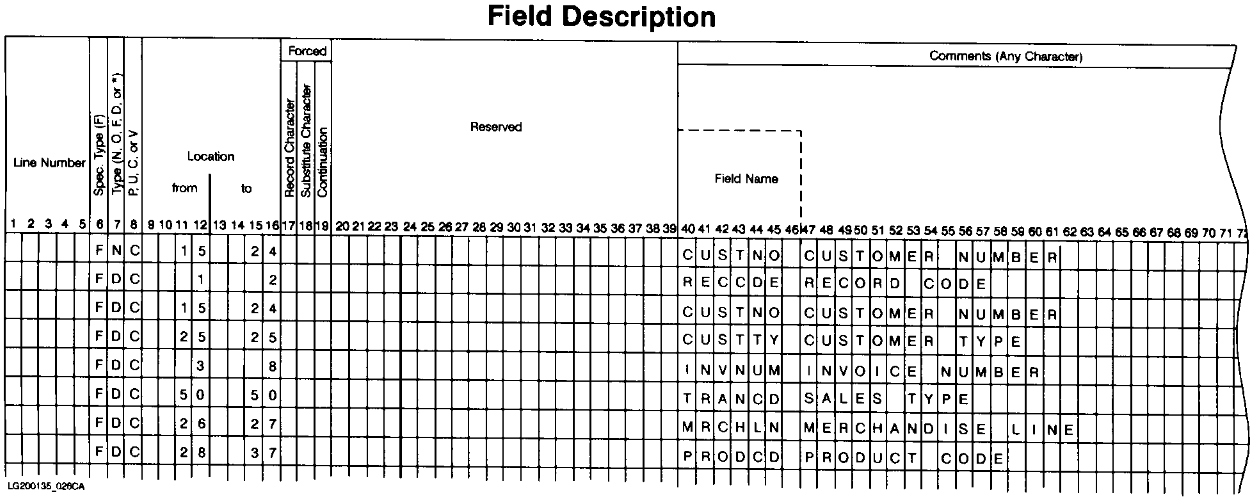
HEWLETT PACKARD

XSORT SPECIFICATIONS

Record Type

Line Number (1-5)	Spec. Type (I or O) (6)	Continuation (A, O, or *) (7)	C, P, or U (8)	Factor 1 Location from (9-12)	Factor 1 Location to (13-16)	Rel. EQ, NE, LT, GT, LE, GE (17-18)	F or C (19)	Factor 2 (Field or Constant) Constant (Any ASCII Character) (20-39)	Record Name (40-48)	Comments (Any Character) (49-71)
	I		C	1	2	EQ	C	WS		SELECT WEEKLY SALCT
	I	*								RECORD

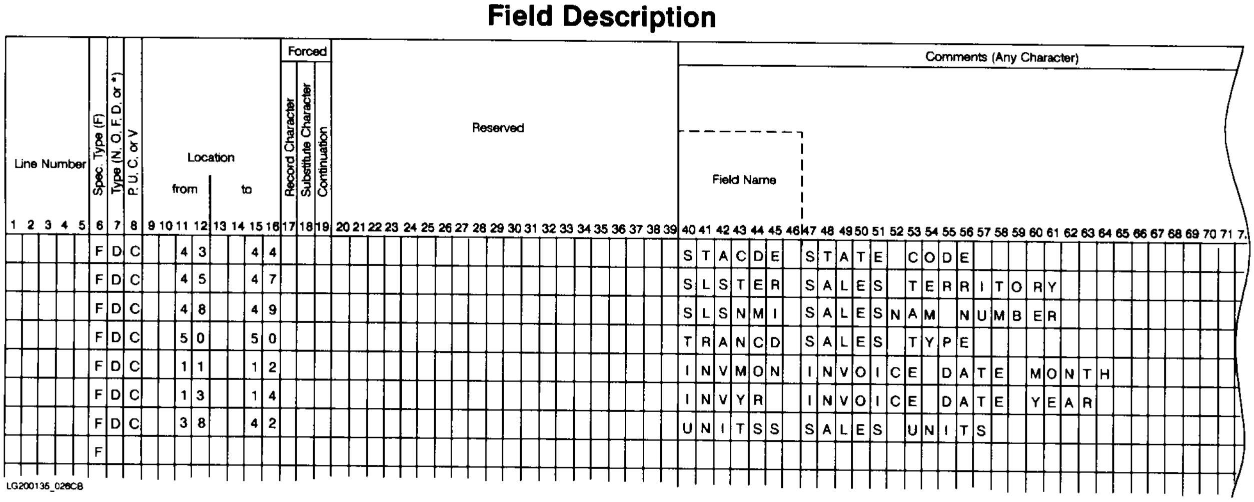
LG200135_026B



Field Description

Line Number (1-5)	Spec. Type (F) (6)	Type (N, O, F, D, or *) (7)	P, U, C, or V (8)	Location from (9-12)	Location to (13-16)	Forced Record Character (17)	Forced Substitute Character (18)	Continuation (19)	Reserved (20-39)	Field Name (40-46)	Comments (Any Character) (47-72)
	F	N	C	15	24					CUSTNO	CUSTOMER NUMBER
	F	D	C	1	2					RECCDE	RECORD CODE
	F	D	C	15	24					CUSTNO	CUSTOMER NUMBER
	F	D	C	25	25					CUSTTY	CUSTOMER TYPE
	F	D	C	3	8					INVNUM	INVOICE NUMBER
	F	D	C	50	50					TRANCD	SALES TYPE
	F	D	C	26	27					MRCHLN	MERCHANDISE LINE
	F	D	C	28	37					PRODCD	PRODUCT CODE

LG200135_026CA



Field Description

Line Number (1-5)	Spec. Type (F) (6)	Type (N, O, F, D, or *) (7)	P, U, C, or V (8)	Location from (9-12)	Location to (13-16)	Forced Record Character (17)	Forced Substitute Character (18)	Continuation (19)	Reserved (20-39)	Field Name (40-46)	Comments (Any Character) (47-72)
	F	D	C	43	44					STACDE	STATE CODE
	F	D	C	45	47					SLSTER	SALES TERRITORY
	F	D	C	48	49					SLSNMI	SALESNAM NUMBER
	F	D	C	50	50					TRANCD	SALES TYPE
	F	D	C	11	12					INVMON	INVOICE DATE MONTH
	F	D	C	13	14					INVYR	INVOICE DATE YEAR
	F	D	C	38	42					UNITSS	SALES UNITS
	F										

LG200135_026CB

Explanation of Specifications.

Header Specification.

Column 6 The H identifies this as the Header line.

Cols 7-12 SORT R identifies this as a Record-out sort.

Cols 13-17 10 is the largest total of control field lengths for either of the two record sections in the sort.

It is the sum derived from adding together all the control fields (fields with N, O, or F in column 7 of the Field Description specifications). In this case, control field length for the first record section is the same as that for the second section since it represents the same field, Customer Number. The field's location changes due to differences in the record formats of the two input files.

Column 18 The A indicates that the records in the sorted file will be in ascending order by control field.

Column 28 The X entered will cause the control fields to be dropped from the output records.

Cols 29-32 90 is the length of the output records in the Record-out sort job. In this sort, control fields are being dropped so the total reflects the largest total of data fields for any single record section. If the control fields were being included in the output file, their largest total (from Header columns 13-17) would be added to that of the data fields.

S Option.

Column 6 The S indicates that the S Option is being used. In this case, it is required because of the multiple input files.

Cols 10-16 36 is the maximum number of records to be sorted.

Record Type Specifications (First Record Section).

Column 6 The I entry identifies this as an include line.

Column 7 Leaving this blank identifies this as the first line of an include set.

Column 8 C means the full character--1 byte (8 bits)--will be used in comparisons done for this sort.

Cols 9-12 1 is the position in the input record where the factor 1 field begins.

Cols 13-16 2 is the position in the input record where the factor 1 field ends.

Cols 17-18 EQ means that factor 1 must equal factor 2 in the comparisons.

Column 19 The C identifies factor 2 as a constant rather than a field.

Cols 20-39 WB is the factor 2 constant.

Field Description Specifications (First Record Section).

Column 6 F indicates that this is a field description specification.

Column 7 N identifies a normal control field. D identifies a data field.

Column 8 C indicates that a full character--1 byte (8 bits)--will be used.

Cols 9-12 3 is the input record position where the control field begins. 1 is the input record position where the data field begins.

Cols 13-16 12 is the input record position where the control field ends. 50 is the input record position where the data field ends.

Record Type Specifications (Second Record Section).

Column 6 I indicates that this is an include line.

Column 7 Leaving this entry blank means that this is the first line of an include set. The * means comments.

Column 8 C means the full character--1 byte (8 bits)--will be used in the comparisons.

Cols 9-12 A 1 entered here indicates the position in the record where the factor 1 field begins.

Cols 13-16 2 entered here indicates the position in the record where the factor 1 field ends.

Cols 17-18 EQ means that, in the comparisons, factor 1 must equal factor 2.

Column 19 The C indicates that factor 2 is a constant, not a field.

Cols 20-27 WS is the factor 2 constant. Note that this is the same length--2 characters--as the factor 1 field.

Field Description Specifications (Second Record Section).

Column 6 F indicates that this is a field description line.

Column 7 N identifies this as a normal con trol field. D identifies this as a data field.

Column 8 C entered here means the full character--1 byte (8 bits)--will be used in the comparisons.

Cols 9-12 These entries identify the position in the input records where control and data fields begin. Note that Customer Number, which is used as a control field, is entered here again as a data field. Since control fields are being dropped from output records, they must be redefined as data fields in order to be included as output.

Cols 13-16 Numbers entered in these columns identify the positions in the input records where the fields end.

This is the output of the sort.

```
WS01111111100900003009VVV0000000CA031100018101234

WB11111111115999995505AAA5555555CA0150510181567895

WB11111111119999989109ZZZ9999999CA31202N0281901233

WS11111111119900012905ZZZ9999999CA011019028112345

WS12111111121900002109ZZZ9999999CA031111018123456

WB21111111128999998205XXX8888888TX0120210281234568

WB21111111124999994605DDD4444444CA0160610281678904

WB21111111120999990009XXX0000000CA21101N0281901232

WS21111111128900011805XXX8888888CA011028028123456

WS23111111132900001209XXX8888888CA031122018134567

WS31111111137900010705LLL7777777TX011037018134567

WB41111111149999999105ZZZ9999999CA0110110181123459

W841111111146999996405VVV6666666NV0140411181234570

WB41111111142999992809ZZZ2222222CA91808N0181890122

WB41111111148999988209ZZZ8888888CA41303N1281901234
```

```
WS41111111146900009605VVV6666666AZ011046018145678

WS51111111155900008505AAA5555555CA011055018156789

WS51111111153900006309JJJ3333333CA031073018178901

WS61111111164900007405DDD4444444CA011064018167890

WB71111111177999997305LLL7777777AZ0130311281234569

WB71111111173999993709LLL3333333CA81707N0281789013

WB71111111171999991909ZZZ1111111CA11909Y0181901231

WS81111111182900005209GGG2222222CA031082018189012

WS91111111191900004109ZZZ1111111CA031091018190123
```

A Appendix A COMPATIBILITY WITH OTHER SYSTEMS

Compatibility with SORT3

XSORT is based on the unsupported utility program named SORT3. To avoid difficulties in converting from SORT3, XSORT has been designed to recognize the formal file designators used by SORT3. There are no significant differences in the syntax of the sort specifications between the two products. The only issue to resolve in the conversion is the naming of the program file--XSORT vs SORT3. To do this, first use the command:

```
:LISTF SORT3.@.@,1
```

This will determine all the groups and accounts in which SORT3 has been stored. Then log on to each of those groups and do the following:

```
:PURGE SORT3
:FCOPY FROM=XSORT.PUB.SYS;TO=SORT3;NEW
```

This will allow all old SORT3 job streams, UDC's, and operator procedures to work without requiring changes for XSORT. The following table shows the correlation between the formal file names for XSORT and SORT3.

File	XSORT NAMES	SORT3 NAMES	CONVERSION ACTIONS
Program	XSORT.PUB.SYS	SORT3.grp.acct	:HELLO user.acct, grp :PURGE SORT3 :FCOPY FROM=XSORT.PUB.SYS;& :TO=SORT3;NEW
Specifications	XSORTEXT	SPECCARD	No action necessary.
Single input	XSORTIN	SORT3IN	No action necessary. (Note, however, that the first input file, "...IN" or "...IN1," is what determines whether XSORT or SORT3 names will be expected for rest of the input/output files. No mixing of formal file names is allowed among the input and output files within any sort job.)
Multiple input	XSORTIN1 : XSORTIN9	SORT3IN1 : SORT3IN9	No action necessary. (Note, however, that the first input file, "...IN" or "...IN1," is what determines whether XSORT or SORT3 names will be expected for rest of the input/output files. No mixing of formal file names is allowed among the input and output files within any sort job.)

File	XSORT NAMES	SORT3 NAMES	CONVERSION ACTIONS
Output	XSORTOUT	SORT3OUT	No action necessary. (Note, however, that the first input file, "...IN" or "...IN1," is what determines whether XSORT or SORT3 names will be expected for rest of the input/output files. No mixing of formal file names is allowed among the input and output files within any sort job.)

Comparison with $DSORT and #GSORT

This section compares XSORT with IBM System/3 $DSORT and IBM System 32 and 34 #GSORT.

XSORT provides:

- Support to handle multiple input files up to 9.
- A "Count-only" pass (SORTC) capability.
- An additional specification record called S Option, identified by an S entry in column 6, which provides additional global controls including:
    - Address-out file first record number.
    - Inplace sort override.
    - Maximum number of records to be sorted.

XSORT does not provide:

- Support for data types "Z" (zone) and "D" (digit) for independent use of EBCDIC zones and digits. XSORT supports data type "C" (complete ASCII character).
- Summary Sort capability.
- The IBM #GSORT reserved "keywords" UDATE, UMONTH, UDAY, and UYEAR.
- The "workfile data verification" feature as controlled by IBM $DSORT and #GSORT through Header Column 34.
- "Null Output Message" option as controlled by IBM #GSORT Header Column 36.
- A 96 column specification form. XSORT's specification form is 80 columns. Users of $DSORT or #GSORT using Alternate Collating Sequences of more than 80 columns will have to adapt them to the 80-column format when converting to XSORT.

B Appendix B ERROR MESSAGES

XSORT error messages distinguish between two types of errors--syntax errors and execution errors.

In the discussion of these messages below, the syntactical errors are covered first. They are broken down by the category under which they can be grouped. General syntactical errors are followed by those which occur under the headings of the various types of specifications: Header, ALTSEQ, S Option, Record Type, and Field Description.

Syntax messages are caused by the detection of some type of inconsistency during the interpretation of the XSORT specifications. They are included in the listings of the specifications on $STDLIST and usually follow immediately the specification section to which they belong--Header, ALTSEQ, S Option, Record Type, or Field Description.

The error number associated with a message is printed on the right hand margin. When an error can be associated with a particular set of columns on a specification line, that set of columns is underlined with asterisks (*****).

On the following pages is a list of errors with their corresponding numbers and further explanations of their meanings when necessary.

Syntax Messages

General Errors

NUMBER	MESSAGE	FURTHER EXPLANATION
202	INVALID SPECIFICATION TYPE-COL 6 MUST BE H, I, O, OR F	Only exceptions are ALTSEQ records.
1001	MORE THAN ONE HEADER SPEC	XSORT specifications must begin with one, and only one, header specification.
1004	BEGIN/END COLUMN MISSING OR INVALID NUMERIC	For both Record Type and Field Description specifications, the "Location From/To" entries in columns 9-16 must contain valid numeric values.
1005	BEGIN COL>END COL OR END COL OUTSIDE OF RECORD	For both Record Type and Field Description specifications, the "Location From" entry in columns 9-12 must not be greater than the "To" entry in columns 13-16. Also, the "To" value must not exceed the actual record length of the XSORTIN file.
1012	BEGIN COL MUST BE BLANK	"From location" (Columns 9-12) must be blank or the same as "To location" (Columns 13-16) for conditionally forced control fields.

Header Errors

NUMBER	MESSAGE	FURTHER EXPLANATION
103	MUST BE SORTA, SORTC, OR SORTR	Columns 7-12 must contain SORTA, SORTC, or SORTR.
151	NO CONTROL FIELD LENGTH SPECIFIED	Columns 13-17 must contain the largest sum of control field lengths of any record section (in bytes)
162	INVALID NUMBER IN OUTPUT RECORD LENGTH FIELD	Columns 29-32 must contain a valid numerical value.
1002	BAD KEY RETENTION OPT--MUST BE BLANK OR X	Column 28 must contain either a blank, to retain control fields on output records, or "X" to drop the control fields.
1003	ASCENDING/DESCENDING OPT INVALID-COL 18 MUST BE A OR D	
	ALTSEQ MUST BE S OR BLANK -COL 26	
1025	PRINT OPTION (COL 27) MUST BE 0, 1, 2, OR 3	

Alternate Collating Sequence Errors

NUMBER	MESSAGE	FURTHER EXPLANATION
1015	ALTSEQ TABLE MUST END WITH **	ALTSEQ specifications must be terminated by a record containing "**" in columns 1-2.
1016	INVALID HEX DIGIT	Each hexidecimal character must be in the range of ASCII characters "O" through "F"
1023	MAX FIELD LENGTH WITH ALTSEQ IS 256 CHARACTERS	When an alternate collating sequence is being used, the maximum length of any control or data field is 256 characters.

S Option Errors

NUMBER	MESSAGE	FURTHER EXPLANATION
1020	MAXIMUM NUMBER OF RECS TO BE SORTED INVALID	Columns 10-16 must be blank or contain a valid, non-negative numeric value.
1021	INVALID EXISTENCE OPTION--COL 8 MUST BE Y, N, OR BLANK	
1022	INVALID FIRST RECMUST BE 0, 1, OR BLANK ORDS#--COL 7	

Record Type Errors

NUMBER	MESSAGE	FURTHER EXPLANATION
206	INVALID CONTINUATION-COL 7 MUST BE A, 0, OR BLANK	
210	INVALID DATA TYPE-COL 8 MUST BE A, C, P, OR U	
212	INVALID SET-OMIT & FIELD DESC BUT NO INCLUDE	Omit sets must always be followed by at least one Include prior to entry of Field Description specs.
214	INCLUDE OR OMIT SPECS AFTER AN INCLUDE ALL	"Include all," if used, must be specified after all Include/Omit sets for this record section.
216	INCOMPLETE SET-INCLUDE OR OMIT W/OUT FOLLOWING FIELD DESC'S	Include/Omit sets must be followed by Field Description specs.
236	INVALID RELATION-COL 17-18 MUST BE EQ, NE, LT, GT, LE, OR GE	
238	INVALID FACTOR 2 TYPE- COL 19 MUST BE C OR F	

NUMBER	MESSAGE	FURTHER EXPLANATION
1019	INVALID DECIMAL DIGIT	Constant in columns 20-80 must contain valid decimal digits and "+" or "-" in last position.
1024	MAX U FIELD LENGTH IS 28 DIGITS	The maximum length of any unpacked numeric field is 28 digits.
1006	FIELD LENGTHS IN COMPARE ARE DIFFERENT	For a field compare ("F" in column 19), Factor 1 (columns 9-16) field length must be the same as Factor 2 (columns 20-27) field length.
1018	PACKED CONSTANT MUST HAVE + OR -SIGN	Constant in columns 20-80 must include a trailing "+" or "-" sign when being compared to a packed data field.

Field Description Errors

NUMBER	MESSAGE	FURTHER EXPLANATION
246	INVALID FIELD TYPE -COL 7 MUST BE N, O, D, F OR *	
252	CONTROL FIELD LENGTH EXCEEDS HEADER VALUE	The sum of control field lengths for this record section exceeds the maximum sum specified in Header columns 13-17.
262	DATA LENGTH EXCEEDS HEADER VALUE	Total length of data in the output record for this record section exceeds the length specified in Header columns 29-32.
1007	DATA TYPE INVALID -COL 8 MUST BE P, U, C, OR V	
1008	RECORD CHAR ONLY LEGAL IN CONDITIONAL FORCE	Record character (column 17) is only used for specification of a conditionally forced control field.
1010	FORCE ALL WITHOUT CONDITIONAL FORCE PRECEDING	A force-all control field must be preceded by at least one conditionally forced control field.
1011	INVALID CONDITIONAL FORCE CONTINUATION	Continuation (column 19) is not valid with: * An unconditionally forced control field. * A conditionally forced control field that immediately follows a one-character normal or opposite control field that is not at the same record location.
1013	FORCE ALL & CONTROL FIELD CONTINUE MUTUALLY EXCLUSIVE	A set of conditionally forced control fields that immediately follow a normal or opposite control field cannot be followed by a force-all control field.
1014	DATA FIELDS NOT ALLOWED WITH ADDRESS-OUT SORT	Data fields ("D" in column 7) cannot be specified with an Address-out sort (SORTA in Header columns 7-11)

Unnumbered Errors

The two unnumbered errors belong in the syntax error group.

MESSAGE	FURTHER EXPLANATION
NO HEADER SPEC	XSORT specifications must begin with one and only one header specification.
NO FIELD DESC SPEC	Field description specifications (formats) must be included for Address-out (SORTA) and Record-out (SORTR) sorts

Execution Messages

Execution messages are caused by some abnormal condition occurring during the actual sort, after the specifications have been analyzed. When one of these errors occurs, the following takes place:

- A "SORTSTONE" is printed showing one of the error messages printed below.

- A "File Information Display" is printed if the error involves a file.
- The sort is aborted. The error number, always preceded by a - (minus sign), appears after the PARAM = in the abort message.

NUMBER	MESSAGE	FURTHER EXPLANATION
-1	ERROR ON XSORTEXT	Error on opening or reading XSORTEXT file.
-2	ERROR ON $STDLIST	Error on print to $STDLIST.
-3	ERROR ON XSORTIN	Error on opening, accessing, or closing XSORTIN file.
-4	ERROR ON XSORTOUT	Error on opening, accessing, or closing XSORTOUT file or XSORT999 file (used for inplace sort).
-5	XSORTOUT EXISTS BUT NOT XSORTIN	If the XSORTOUT file already exists, it must be the same as XSORTIN. (This restriction can be overridden by specifying "Y" in S Option column 8).
-6	XSORTIN RECORD BIGGER THAN BUFFER-CHANGE MAX'IN'REC	Record length of XSORTIN file exceeds 1000 characters. MAX'IN'REC is a value hard-coded in XSORT source code and so can only be changed by recompilation of XSORT by HP.
-10	TERMINAL ERROR IN SPECS- SEE LISTING	One or more syntax errors occurred in the specifications. See the listings for the specific message.
-11	OUT OF TABLE SPACE-IBUF TOO SMALL	Not enough room in the internal routine table for all Record and Field specifications. The number of specifications must be reduced to fit within te 8192 character internal table. The amount of space| (in characters) required in the table can be calculated by the following formula: 30 + 12 * (number of Record and Field Specs) + (sum of lengths of constants on Record Specs + 512 (if ALTSEQ is used)
-12	SORTINITAL FAILED	Internal error: use of SORTINITAL intrinsic failed.
-13	SORTINPUT FAILED	Internal error: use of SORTINPUT intrinsic failed.
-14	SORTOUTPUT FAILED	Internal error: use of SORTOUTPUT intrinsic failed.
-15	ATTEMPTED TO OUTPUT AN RSAM FILE	Inplace sort not allowed when XSORTIN is an RSAM file.
-16	XSORTOUT RECORD SIZE TOO SMALL	Required output record size (computed from specifications) exceeds the actual record size of XSORTOUT.
-17	WORK BUFFER TOO SMALL CHANGE MAX'WORK'REC	Work record (control fields and data fields) cannot exceed 1000 characters. MAX'WORK'REC is a value hardcoded in XSORT source code and so can only be changed by modification and recompilation of XSORT by HP.

NUMBER	MESSAGE	FURTHER EXPLANATION
-18	LOADPROC OF FREADC FAILED	Internal error: loading procedure for FREADC intrinsic failed.
-19	MULTI INPUT (XSORTIN1) AND SORTA INVALID	Address-out sort (SORTA in Header columns 7-11 not allowed when multiple input files (XSORTIN1.....XSORTIN9) are used.
-20	MULTI INPUT AND INPLACE SORT INVALID	Inplace sort not allowed when multiple input files (XSORTIN1.....XSORTIN9) are used.
-21	CAN'T SORT KSAM FILE INPLACE	Inplace sort not allowed when XSORTIN is a KSAM file.

C Appendix C DATA INFORMATION

Collating Sequence Table

The Collating Sequence Table brings together comparative information concerning data used in sort jobs. It includes the ASCII and EBCDIC collating sequences in decimal, octal, hexidecimal, and binary representations. It gives the character (graphic) or control equivalents for both ASCII and EBCDIC.

The relative position of alphabetic and numeric characters in the sorting sequence is shown in the right hand columns. This information will be helpful to those users who have experience with systems based on the EBCDIC collating sequence. A major difference between the two systems is the relative position of alphabetic and numeric characters in the collating sequence. When alphabetic characters have a higher place in the collating sequence than numeric characters, this can effect compare and matching field operations. Using Alternate Collating Sequence, it is possible to reverse this if the user prefers EBCDIC sequence..

COLLATING SEQUENCE					ALPHA/CONTROL CHARACTER		NUMERIC VALUE	
Dec	**Oct**	**Hex**	**Binary**		**ASCII**	**EBCDIC**	**ASCII**	**EBCDIC**
0	000	00	0000	0000	NUL	NUL		
1	001	01	0000	0001	SOH	SOH		
2	002	02	0000	0010	STX	STX		
3	003	03	0000	0011	ETX	ETX		
4	004	04	0000	0100	EOT	PF		
5	005	05	0000	0101	ENQ	HT		
6	006	06	0000	0110	ACK	LC		
7	007	07	0000	0111	BEL	DEL		
8	010	08	0000	1000	BS			
9	011	09	0000	1001	HT			
10	012	0A	0000	1010	LF	SMM		
11	013	0B	0000	1011	VT	VT		
12	014	0C	0000	1100	FF	FF		
13	015	0D	0000	1101	CR	CR		
14	016	0E	0000	1110	SO	SO		
15	017	0F	0000	1111	SI	SI		
16	020	10	0001	0000	DLE	DLE		

COLLATING SEQUENCE					ALPHA/CONTROL CHARACTER		NUMERIC VALUE	
Dec	Oct	Hex	Binary		ASCII	EBCDIC	ASCII	EBCDIC
17	021	11	0001	0001	DC1	DC1		
18	022	12	0001	0010	DC2	DC2		
19	023	13	0001	0011	DC3	TM		
20	024	14	0001	0100	DC4	RES		
21	025	15	0001	0101	NAK	NL		
22	026	16	0001	0110	SYN	BS		
23	027	17	0001	0111	ETB	IL		
24	030	18	0001	1000	CAN	CAN		
25	031	19	0001	1001	EM	EM		
26	032	1A	0001	1010	SUB	CC		
27	033	1B	0001	1011	ESC	CU1		
28	034	1C	0001	1100	FS	IFS		
29	035	1D	0001	1101	GS	IGS		
30	036	1E	0001	1110	RS	IRS		
31	037	1F	0001	1111	US	IUS		
32	040	20	0010	0000	SP	DS		
33	041	21	0010	0001	!	SOS		
34	042	22	0010	0010	"	FS		
35	043	23	0010	0011	#			
36	044	24	0010	0100	$	BYP		
37	045	25	0010	0101	%	LF		
38	046	26	0010	0110	&	ETB		
39	047	27	0010	0111	'	ESC		
40	050	28	0010	1000	(			
41	051	29	0010	1001	)			
42	052	2A	0010	1010	*	SM		
43	053	2B	0010	1011	+	CU2		
44	054	2C	0010	1100	,			

COLLATING SEQUENCE					ALPHA/CONTROL CHARACTER		NUMERIC VALUE	
Dec	Oct	Hex	Binary		ASCII	EBCDIC	ASCII	EBCDIC
45	055	2D	0010	1101	-	ENQ		
46	056	2E	0010	1110	.	ACK		
47	057	2F	0010	1111	/	BEL		
48	060	30	0011	0000	0		0	
49	061	31	0011	0001	1		1	
50	062	32	0011	0010	2	SYN	2	
51	063	33	0011	0011	3		3	
52	064	34	0011	0100	4	PN	4	
53	065	35	0011	0101	5	RS	5	
54	066	36	0011	0110	6	UC	6	
55	067	37	0011	0111	7	EOT	7	
56	070	38	0011	1000	8		8	
57	071	39	0011	1001	9		9	
58	072	3A	0011	1010	:			
59	073	3B	0011	1011	;	CU3		
60	074	3C	0011	1100	<	DC4		
61	075	3D	0011	1101	=	NAK		
62	076	3E	0011	1110	>			
63	077	3F	0011	1111	?	SUB		
64	100	40	0100	0000	@	SP		
65	101	41	0100	0001	A		+1	
66	102	42	0100	0010	B		+2	
67	103	43	0100	0011	C		+3	
68	104	44	0100	0100	D		+4	
69	105	45	0100	0101	E		+5	
70	106	46	0100	0110	F		+6	
71	107	47	0100	0111	G		+7	
72	110	48	0100	1000	H		+8	

COLLATING SEQUENCE					ALPHA/CONTROL CHARACTER		NUMERIC VALUE	
Dec	Oct	Hex	Binary		ASCII	EBCDIC	ASCII	EBCDIC
73	111	49	0100	1001	I		+9	
74	112	4A	0100	1010	J	Â	-1	
75	113	4B	0100	1011	K	.	-2	
76	114	4C	0100	1100	L	<	-3	
77	115	4D	0100	1101	M	(	-4	
78	116	4E	0100	1110	N	+	-5	
79	117	4F	0100	1111	O			-6
80	120	50	0101	0000	P	&	-7	
81	121	51	0101	0001	Q		-8	
82	122	52	0101	0010	R		-9	
83	123	53	0101	0011	S			
84	124	54	0101	0100	T			
85	125	55	0101	0101	U			
86	126	56	0101	0110	V			
87	127	57	0101	0111	W			
88	130	58	0101	1000	X			
89	131	59	0101	1001	Y			
90	132	5A	0101	1010	Z	!		
91	133	5B	0101	1011	[	$		
92	134	5C	0101	1100	\	*		
93	135	5D	0101	1101	]	)		
94	136	5E	0101	1110	\	;		
95	137	5F	0101	1111	...	-		
96	140	60	0110	0000		-		
97	141	61	0110	0001	a	/		
98	142	62	0110	0010	b			
99	143	63	0110	0011	c			
100	144	64	0110	0100	d			

COLLATING SEQUENCE					ALPHA/CONTROL CHARACTER		NUMERIC VALUE	
Dec	Oct	Hex	Binary		ASCII	EBCDIC	ASCII	EBCDIC
101	145	65	0110	0101	e			
102	146	66	0110	0110	f			
103	147	67	0110	0111	g			
104	150	68	0110	1000	h			
105	151	69	0110	1001	i			
106	152	6A	0110	1010	j			
107	153	6B	0110	1011	k	,		
108	154	6C	0110	1100	l	%		
109	155	6D	0110	1101	m	-		
110	156	6E	0110	1110	n	>		
111	157	6F	0110	1111	o	?		
112	160	70	0111	0000	p			
113	161	71	0111	0001	q			
114	162	72	0111	0010	r			
115	163	73	0111	0011	s			
116	164	74	0111	0100	t			
117	165	75	0111	0101	u			
118	166	76	0111	0110	v			
119	167	77	0111	0111	w			
120	170	78	0111	1000	x			
121	171	79	0111	1001	y	‘		
122	172	7A	0111	1010	z	:		
123	173	7B	0111	1011	{	#	+0	
124	174	7C	0111	1100			@	
125	175	7D	0111	1101	}	'	-0	
126	176	7E	0111	1110	~	=		
127	177	7F	0111	1111	DEL	"		
128	200	80	1000	0000				

COLLATING SEQUENCE					ALPHA/CONTROL CHARACTER		NUMERIC VALUE	
Dec	Oct	Hex	Binary		ASCII	EBCDIC	ASCII	EBCDIC
129	201	81	1000	0001		a		
130	202	82	1000	0010		b		
131	203	83	1000	0011		c		
132	204	84	1000	0100		d		
133	205	85	1000	0101		e		
134	206	86	1000	0110		f		
135	207	87	1000	0111		g		
136	210	88	1000	1000		h		
137	211	89	1000	1001		i		
138	212	8A	1000	1010				
139	213	8B	1000	1011				
140	214	8C	1000	1100				
141	215	8D	1000	1101				
142	216	8E	1000	1110				
143	217	8F	1000	1111				
144	220	90	1001	0000				
145	221	91	1001	0001		j		
146	222	92	1001	0010		k		
147	223	93	1001	0011		l		
148	224	94	1001	0100		m		
149	225	95	1001	0101		n		
150	226	96	1001	0110		o		
151	227	97	1001	0111		p		
152	230	98	1001	1000		q		
153	231	99	1001	1001		r		
154	232	9A	1001	1010				
155	233	9B	1001	1011				
156	234	9C	1001	1100				

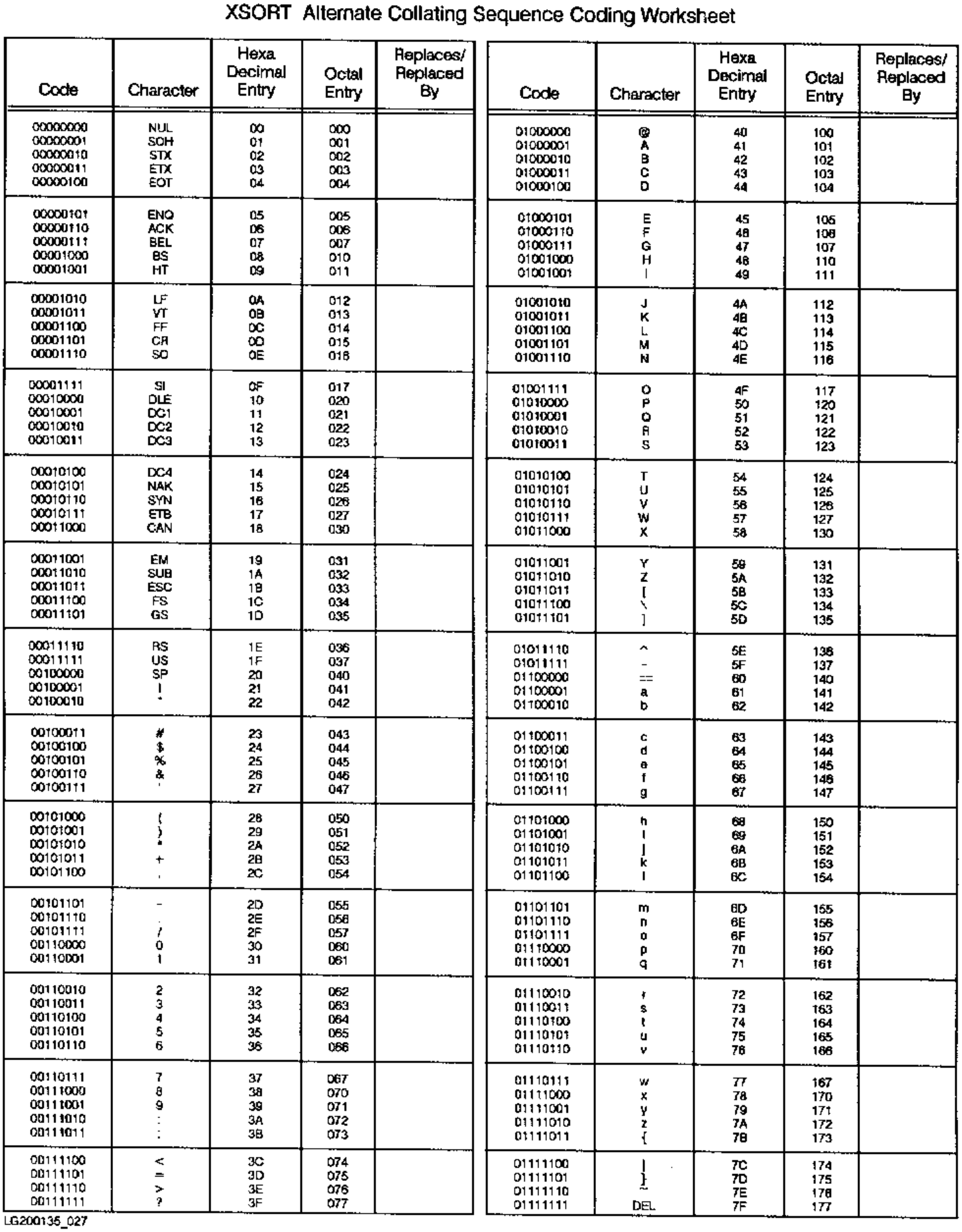
COLLATING SEQUENCE					ALPHA/CONTROL CHARACTER		NUMERIC VALUE	
Dec	Oct	Hex	Binary		ASCII	EBCDIC	ASCII	EBCDIC
157	235	9D	1001	1101				
158	236	9E	1001	1110				
159	237	9F	1001	1111				
160	240	A0	1010	0000				
161	241	A1	1010	0001		~		
162	242	A2	1010	0010		s		
163	243	A3	1010	0011		t		
164	244	A4	1010	0100		u		
165	245	A5	1010	0101		v		
166	246	A6	1010	0110		w		
167	247	A7	1010	0111		x		
168	250	A8	1010	1000		y		
169	251	A9	1010	1001		z		
170	252	AA	1010	1010				
171	253	A8	1010	1011				
172	254	AC	1010	1100				
173	255	AD	1010	1101				
174	256	AE	1010	1110				
175	257	AF	1010	1111				
176	260	B0	1011	0000				
177	261	B1	1011	0001				
178	262	B2	1011	0010				
179	263	B3	1011	0011				
180	264	B4	1011	0100				
181	265	B5	1011	0101				
182	266	B6	1011	0110				
183	267	B7	1011	0111				
184	270	B8	1011	1000				

COLLATING SEQUENCE					ALPHA/CONTROL CHARACTER		NUMERIC VALUE	
Dec	Oct	Hex	Binary		ASCII	EBCDIC	ASCII	EBCDIC
185	271	B9	1011	1001				
186	272	BA	1011	1010				
187	273	BB	1011	1011				
188	274	BC	1011	1100				
189	275	BD	1011	1101				
190	276	BE	1011	1110				
191	277	BF	1011	1111				
192	300	C0	1100	0000		{		+0
193	301	C1	1100	0001		A		+1
194	302	C2	1100	0010		B		+2
195	303	C3	1100	0011		C		+3
196	304	C4	1100	0100		D		+4
197	305	C5	1100	0101		E		+5
198	306	C6	1100	0110		F		+6
199	307	C7	1100	0111		G		+7
200	310	C8	1100	1000		H		+8
201	311	C9	1100	1001		I		+9
202	312	CA	1100	1010				
203	313	CB	1100	1011				
204	314	CC	1100	1100				
205	315	CD	1100	1101				
206	316	CE	1100	1110				
207	317	CF	1100	1111				
208	320	D0	1101	0000		}		-0
209	321	D1	1101	0001		J		-1
210	322	D2	1101	0010		K		-2
211	323	D3	1101	0011		L		-3
212	324	D4	1101	0100		M		-4

COLLATING SEQUENCE					ALPHA/CONTROL CHARACTER		NUMERIC VALUE	
Dec	Oct	Hex	Binary		ASCII	EBCDIC	ASCII	EBCDIC
213	325	D5	1101	0101		N		-5
214	326	D6	1101	0110		O		-6
215	327	D7	1101	0111		P		-7
216	330	D8	1101	1000		Q		-8
217	331	D9	1101	1001		R		-9
218	332	DA	1101	1010				
219	333	DB	1101	1011				
220	334	DC	1101	1100				
221	335	DD	1101	1101				
222	336	DE	1101	1110				
223	337	DF	1101	1111				
224	340	E0	1110	0000		\		
225	341	E1	1110	0001				
226	342	E2	1110	0010		S		
227	343	E3	1110	0011		T		
228	344	E4	1110	0100		U		
229	345	E5	1110	0101		V		
230	346	E6	1110	0110		W		
231	347	E7	1110	0111		X		
232	350	E8	1110	1000		Y		
233	351	E9	1110	1001		Z		
234	352	EA	1110	1010				
235	353	EB	1110	1011				
236	354	EC	1110	1100				
237	355	ED	1110	1101				
238	356	EE	1110	1110				
239	357	EF	1110	1111				
240	360	F0	1111	0000		0		0

COLLATING SEQUENCE					ALPHA/CONTROL CHARACTER		NUMERIC VALUE	
Dec	Oct	Hex	Binary		ASCII	EBCDIC	ASCII	EBCDIC
241	361	F1	1111	0001		1		1
242	362	F2	1111	0010		2		2
243	363	F3	1111	0011		3		3
244	364	F4	1111	0100		4		4
245	365	F5	1111	0101		5		5
246	366	F6	1111	0110		6		6
247	367	F7	1111	0111		7		7
248	370	F8	1111	1000		8		8
249	371	F9	1111	1001		9		9
250	372	FA	1111	1010				
251	373	FB	1111	1011				
252	374	FC	1111	1100				
253	375	FD	1111	1101				
254	376	FE	1111	1110				
255	377	FF	1111	1111				

[ERROR:] Click here to view figure.



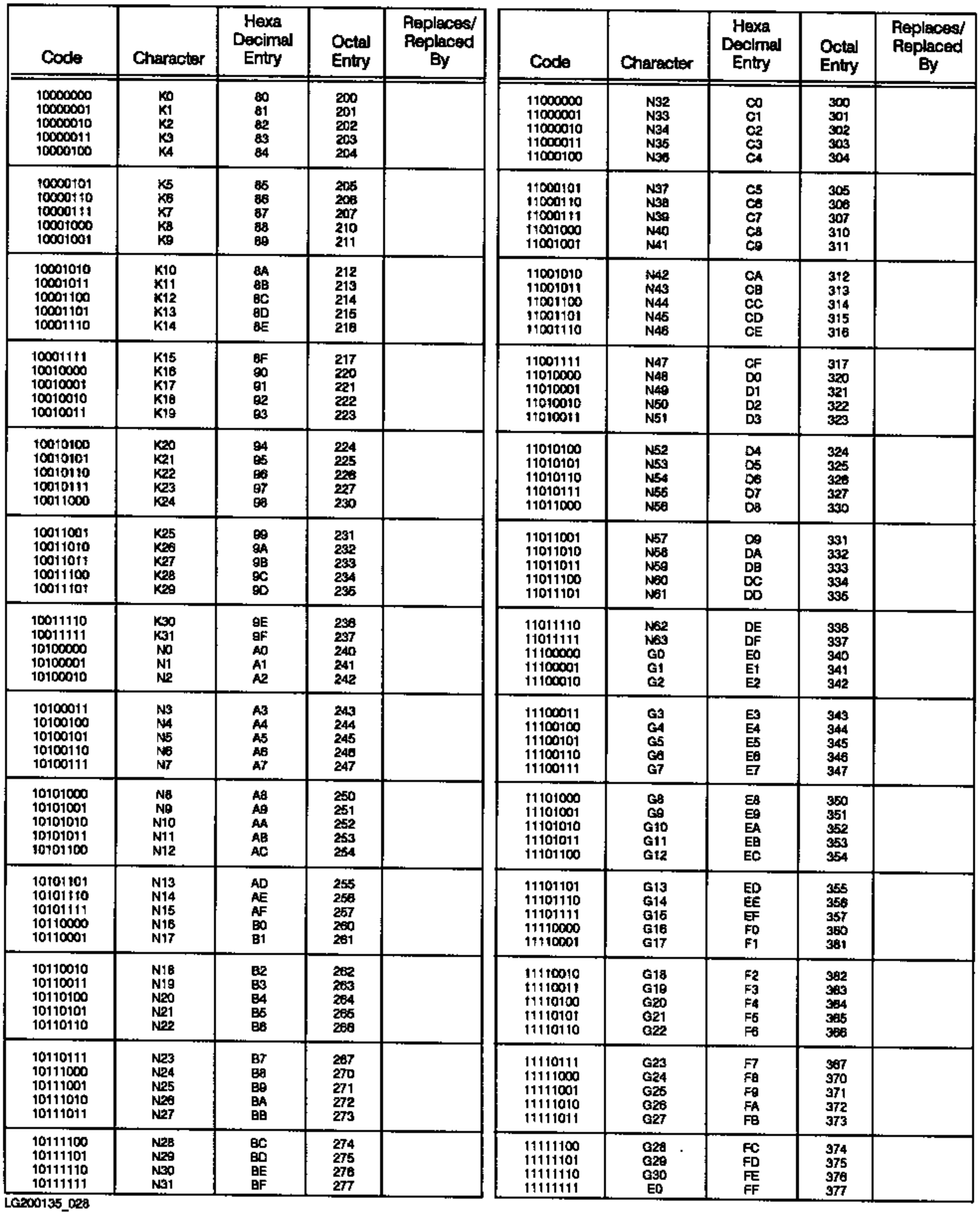
XSORT Alternate Collating Sequence Coding Worksheet

Code	Character	Hexa Decimal Entry	Octal Entry	Replaces/ Replaced By
00000000	NUL	00	000	
00000001	SOH	01	001	
00000010	STX	02	002	
00000011	ETX	03	003	
00000100	EOT	04	004	
00000101	ENQ	05	005	
00000110	ACK	06	006	
00000111	BEL	07	007	
00001000	BS	08	010	
00001001	HT	09	011	
00001010	LF	0A	012	
00001011	VT	0B	013	
00001100	FF	0C	014	
00001101	CR	0D	015	
00001110	SO	0E	016	
00001111	SI	0F	017	
00010000	DLE	10	020	
00010001	DC1	11	021	
00010010	DC2	12	022	
00010011	DC3	13	023	
00010100	DC4	14	024	
00010101	NAK	15	025	
00010110	SYN	16	026	
00010111	ETB	17	027	
00011000	CAN	18	030	
00011001	EM	19	031	
00011010	SUB	1A	032	
00011011	ESC	1B	033	
00011100	FS	1C	034	
00011101	GS	1D	035	
00011110	RS	1E	036	
00011111	US	1F	037	
00100000	SP	20	040	
00100001	!	21	041	
00100010	"	22	042	
00100011	#	23	043	
00100100	$	24	044	
00100101	%	25	045	
00100110	&	26	046	
00100111	'	27	047	
00101000	(	28	050	
00101001	)	29	051	
00101010	*	2A	052	
00101011	+	2B	053	
00101100	,	2C	054	
00101101	-	2D	055	
00101110	.	2E	056	
00101111	/	2F	057	
00110000	0	30	060	
00110001	1	31	061	
00110010	2	32	062	
00110011	3	33	063	
00110100	4	34	064	
00110101	5	35	065	
00110110	6	36	066	
00110111	7	37	067	
00111000	8	38	070	
00111001	9	39	071	
00111010	:	3A	072	
00111011	;	3B	073	
00111100	<	3C	074	
00111101	=	3D	075	
00111110	>	3E	076	
00111111	?	3F	077	
01000000	@	40	100	
01000001	A	41	101	
01000010	B	42	102	
01000011	C	43	103	
01000100	D	44	104	
01000101	E	45	105	
01000110	F	46	106	
01000111	G	47	107	
01001000	H	48	110	
01001001	I	49	111	
01001010	J	4A	112	
01001011	K	4B	113	
01001100	L	4C	114	
01001101	M	4D	115	
01001110	N	4E	116	
01001111	O	4F	117	
01010000	P	50	120	
01010001	Q	51	121	
01010010	R	52	122	
01010011	S	53	123	
01010100	T	54	124	
01010101	U	55	125	
01010110	V	56	126	
01010111	W	57	127	
01011000	X	58	130	
01011001	Y	59	131	
01011010	Z	5A	132	
01011011	[	5B	133	
01011100	\	5C	134	
01011101	]	5D	135	
01011110	^	5E	136	
01011111	-	5F	137	
01100000	==	60	140	
01100001	a	61	141	
01100010	b	62	142	
01100011	c	63	143	
01100100	d	64	144	
01100101	e	65	145	
01100110	f	66	146	
01100111	g	67	147	
01101000	h	68	150	
01101001	i	69	151	
01101010	j	6A	152	
01101011	k	6B	153	
01101100	l	6C	154	
01101101	m	6D	155	
01101110	n	6E	156	
01101111	o	6F	157	
01110000	p	70	160	
01110001	q	71	161	
01110010	r	72	162	
01110011	s	73	163	
01110100	t	74	164	
01110101	u	75	165	
01110110	v	76	166	
01110111	w	77	167	
01111000	x	78	170	
01111001	y	79	171	
01111010	z	7A	172	
01111011	{	7B	173	
01111100	|	7C	174	
01111101	}	7D	175	
01111110	~	7E	176	
01111111	DEL	7F	177	

LG200135_027

[ERROR:] Click here to view figure.

XSORT Alternate Collating Sequence Coding Worksheet



Code	Character	Hexa Decimal Entry	Octal Entry	Replaces/ Replaced By
10000000	K0	80	200	
10000001	K1	81	201	
10000010	K2	82	202	
10000011	K3	83	203	
10000100	K4	84	204	
10000101	K5	85	205	
10000110	K6	86	206	
10000111	K7	87	207	
10001000	K8	88	210	
10001001	K9	89	211	
10001010	K10	8A	212	
10001011	K11	8B	213	
10001100	K12	8C	214	
10001101	K13	8D	215	
10001110	K14	8E	216	
10001111	K15	8F	217	
10010000	K16	90	220	
10010001	K17	91	221	
10010010	K18	92	222	
10010011	K19	93	223	
10010100	K20	94	224	
10010101	K21	95	225	
10010110	K22	96	226	
10010111	K23	97	227	
10011000	K24	98	230	
10011001	K25	99	231	
10011010	K26	9A	232	
10011011	K27	9B	233	
10011100	K28	9C	234	
10011101	K29	9D	235	
10011110	K30	9E	236	
10011111	K31	9F	237	
10100000	N0	A0	240	
10100001	N1	A1	241	
10100010	N2	A2	242	
10100011	N3	A3	243	
10100100	N4	A4	244	
10100101	N5	A5	245	
10100110	N6	A6	246	
10100111	N7	A7	247	
10101000	N8	A8	250	
10101001	N9	A9	251	
10101010	N10	AA	252	
10101011	N11	AB	253	
10101100	N12	AC	254	
10101101	N13	AD	255	
10101110	N14	AE	256	
10101111	N15	AF	257	
10110000	N16	B0	260	
10110001	N17	B1	261	
10110010	N18	B2	262	
10110011	N19	B3	263	
10110100	N20	B4	264	
10110101	N21	B5	265	
10110110	N22	B6	266	
10110111	N23	B7	267	
10111000	N24	B8	270	
10111001	N25	B9	271	
10111010	N26	BA	272	
10111011	N27	BB	273	
10111100	N28	BC	274	
10111101	N29	BD	275	
10111110	N30	BE	276	
10111111	N31	BF	277	
11000000	N32	C0	300	
11000001	N33	C1	301	
11000010	N34	C2	302	
11000011	N35	C3	303	
11000100	N36	C4	304	
11000101	N37	C5	305	
11000110	N38	C6	306	
11000111	N39	C7	307	
11001000	N40	C8	310	
11001001	N41	C9	311	
11001010	N42	CA	312	
11001011	N43	CB	313	
11001100	N44	CC	314	
11001101	N45	CD	315	
11001110	N46	CE	316	
11001111	N47	CF	317	
11010000	N48	D0	320	
11010001	N49	D1	321	
11010010	N50	D2	322	
11010011	N51	D3	323	
11010100	N52	D4	324	
11010101	N53	D5	325	
11010110	N54	D6	326	
11010111	N55	D7	327	
11011000	N56	D8	330	
11011001	N57	D9	331	
11011010	N58	DA	332	
11011011	N59	DB	333	
11011100	N60	DC	334	
11011101	N61	DD	335	
11011110	N62	DE	336	
11011111	N63	DF	337	
11100000	G0	E0	340	
11100001	G1	E1	341	
11100010	G2	E2	342	
11100011	G3	E3	343	
11100100	G4	E4	344	
11100101	G5	E5	345	
11100110	G6	E6	346	
11100111	G7	E7	347	
11101000	G8	E8	350	
11101001	G9	E9	351	
11101010	G10	EA	352	
11101011	G11	EB	353	
11101100	G12	EC	354	
11101101	G13	ED	355	
11101110	G14	EE	356	
11101111	G15	EF	357	
11110000	G16	F0	360	
11110001	G17	F1	361	
11110010	G18	F2	362	
11110011	G19	F3	363	
11110100	G20	F4	364	
11110101	G21	F5	365	
11110110	G22	F6	366	
11110111	G23	F7	367	
11111000	G24	F8	370	
11111001	G25	F9	371	
11111010	G26	FA	372	
11111011	G27	FB	373	
11111100	G28	FC	374	
11111101	G29	FD	375	
11111110	G30	FE	376	
11111111	E0	FF	377	

LG200135_028

D Appendix D XSORT SPECIFICATION FORMS

The forms for specifications are for your convenience. Copies may be made for use in your sort jobs. Hewlett-Packard holds the copyright for the forms and asks that they be copied only for your particular applications.

HEWLETT PACKARD

XSORT SPECIFICATIONS

Header

Line Number	Spec Type (H)	Type of Sort SORTA, SORTR or SORTC	Largest sum of control field lengths of any record section (in bytes)	Sequence (A or D)	Reserved	Alternate Col. Seq. (S)	Print Option	Output Option (X)	Length of output record	Reserved	Comments (any character)
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 68 69 70 71 72
	H										

← Insert ALTSEQ specifications here (if used)

← Insert S OPTION specifications here (if used)

LG200135_029A

Record Type

Line Number	Spec Type (I or O)	Continuation (A, O or *)	C, P or U	Factor 1 Location from	 to	Re EQ, NE, LT, GT, LE, GE	F or C	Factor 2 (Field or Constant) Constant (Any ASCII Character) Location from	 to		Comments (Any Character) Record Name	
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 68 69 70 71 72

LG200135_029B

Field Description

Line Number	Spec Type (F)	Type (N, O, F, D or *)	P, U, C, or V	Location from	 to	Forced Record Character	 Substitute Character	 Continuation	Reserved	Comments (Any Character) Field Name	
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 68 69 70 71 72
	F										
	F										
	F										
	F										
	F										

LG200135_029C

Alternate Collating Sequence

A	L	T	S	E	Q	Blank		← Hexadecimal equivalents of paired characters →
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 68 69 70 71 72

* * (Asterisks in columns 1 and 2 after last ALTSEQ line)

LG200135_037E

S Option

Line Number	Spec. Type (S)	0 or 1	N or Y	Reserved	Maximum number of records to be sorted	Reserved	Comments (Any Character)
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 68 69 70 71 72
	S						

LG200135_037D

HP RPG/XL Utilities-Part 2 RISE

9 INTRODUCTION

The RPG Interactive System Environment is a specialized editor, designed to create and modify programs written in the column-dependent language, RPG.

It is an RPG Utility and, therefore, intended for the use of RPG programmers. You may need it for a simple, quick program, or you may have a job which requires interfacing with KSAM, Image, and VPLUS. However extensive your experience, however technical the job at hand, this system environment will prove a useful, productive, and convenient tool for your programming needs. Throughout this manual, the RPG Interactive System Environment will be referred to as RISE for the sake of simplicity.

RISE Features:

* Visual editing with images that represent the RPG specification coding forms.

* Menu and special function keys (softkeys) for a broad range of uses.

* Page-at-a-time direct screen editing via the terminal's cursor control keys.

* Ability to call the RPG compiler as well as the Segmenter and manipulate the compiled listing using a split screen.

* Choice of editing a file directly or editing a copy of that file.

* Ability to execute important MPE commands and run any program file.

* HELP facility which presents explanations of individual commands or a brief description of all commands.

* Ability to create a comment banner which is presented visually apart from the rest of the source program in which you type your comments.

* Renumbering command which also allows you to renumber the lines (columns 1-5) of the RPG source program.

* Ability to execute commands stored in a file.

* ADD mode which allows you to change RPG specification forms, and to DELETE, LIST, or MODIFY lines while in ADD mode.

* Recoverability features including an undelete command and permanent work files which are protected from system failure

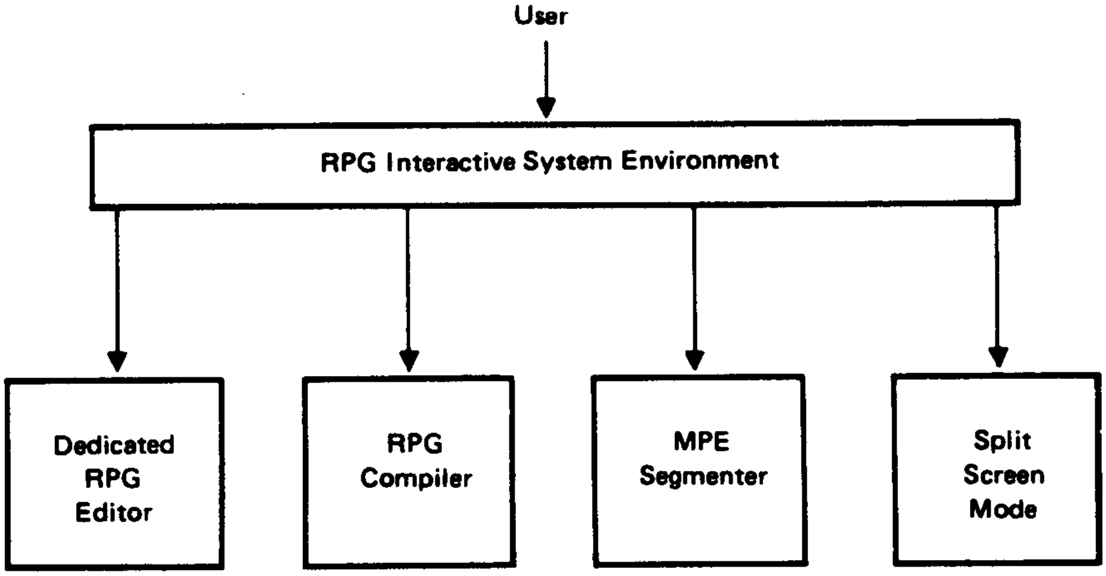


Figure 1-1. RISE environment.

Requirements for Operation

Hardware Requirements

RISE operates as a member of the VPLUS family. Your terminal must be oneof those compatible with VPLUS, such as the model 2645.

In addition, your terminal must have at least 8K of memory and the display enhancement character set. VPLUS requires a minimum of 8K ofmemory in your terminal to store and display RISE's special forms of the RPG Specification Records. These forms utilize the display enhancement character set to the user's advantage.

Software Requirements

Software required to run RISE consists of MPE IV, V, and KSAM. RISE uses VPLUS for its user interface and KSAM for internal management of the file you are editing.

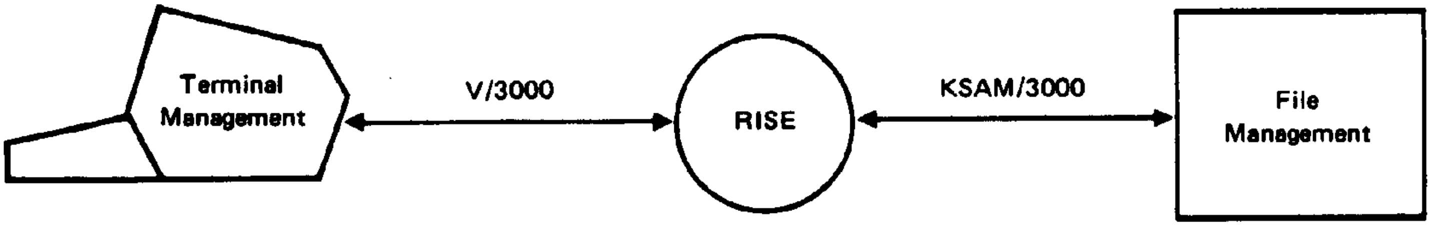


Figure 1-2. Internal/External management.

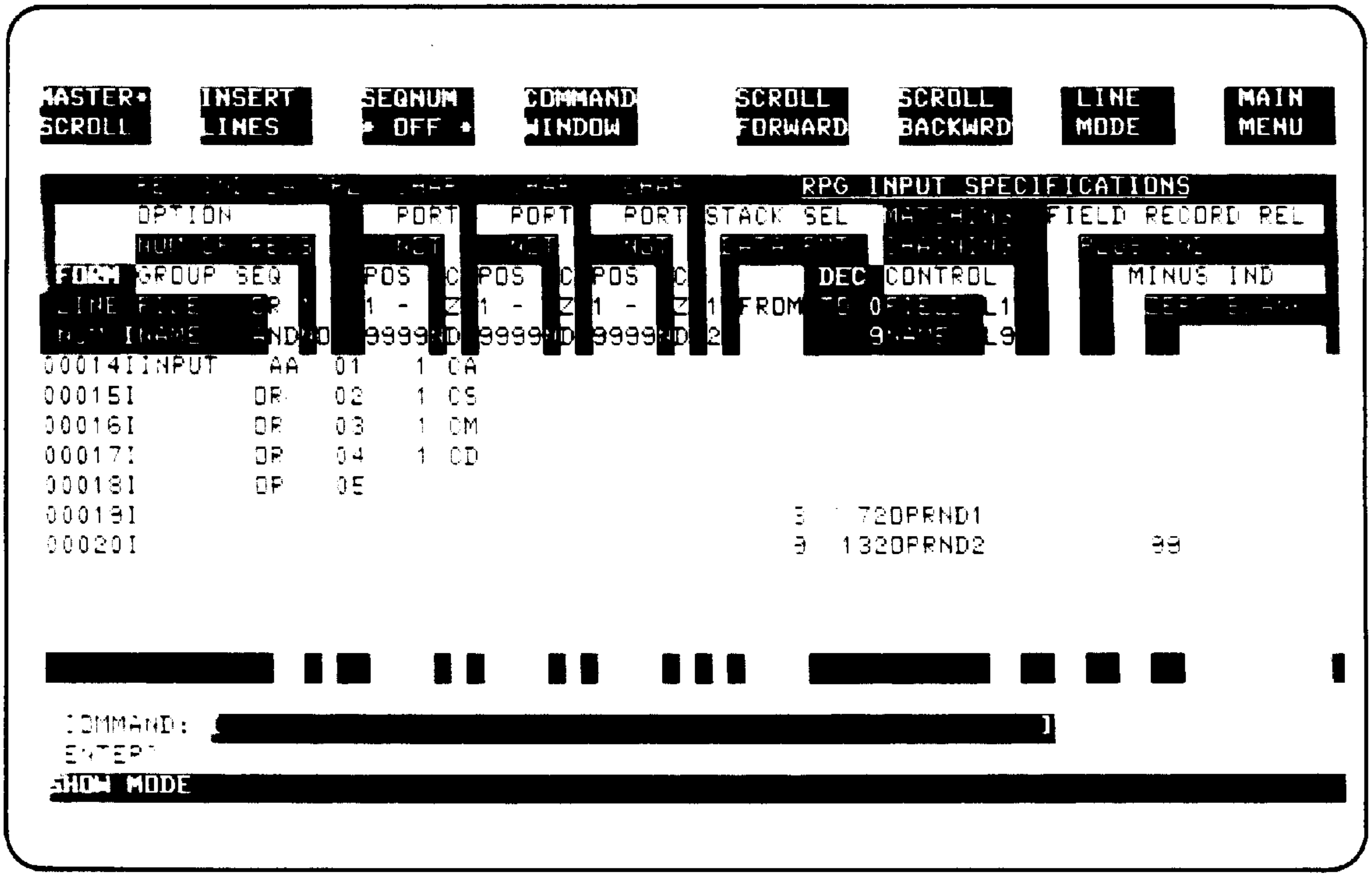
User Interface with Rise

Friendly interface was a guiding principle in the design of RISE. The stress on visual integration of the RPG specification forms makes it much simpler to enter RPG specifications into the system.

Initially, RPG programs are written out on specification coding forms. This allows the user to review the semantics of each column and ensure that they contain the entries necessary to accomplish their data processing tasks and produce the output desired. Entering specifications without a form display can be tedious and confusing, and may require extensive column counting.

RISE allows you to place the appropriate form on the screen for the entry of associated specifications. Protected areas of the screen--those which cannot be changed or modified by the user--contain information and graphic guides which make entering more convenient.

The RPG specification forms available for display are Header Specifications, File Specifications, File Extension Specifications, Line Counter Specifications, Input Specifications, Calculation Specifications, and Output Specifications. You can also use a column numbering display known as a column indicator. Using these forms you can edit, add to, or remove specifications directly on the displayed form.



```
                                                       RPG INPUT SPECIFICATIONS
      OPTION              PORT     PORT     PORT  STACK SEL            FIELD RECORD REL
FORM  GROUP SEQ        POS   C POS   C POS   C            DEC CONTROL       MINUS IND
                OR     1 -   Z 1 -   Z 1 -   Z 1 FROM     0           L1
                 ND    9999 ND 9999 ND 9999 ND 2          9           L9
00014IINPUT     AA  01   1 CA
00015I         OR   02   1 CS
00016I         OR   03   1 CM
00017I         OR   04   1 CD
00018I         OR   05
00019I                                                3   720PRND1
00020I                                                3  1320PRND2          99

COMMAND:
ENTER
SHOW MODE
```

Modes of Operation

Line and Block Modes

RISE operates in Line or Block mode using the terminal's powerful capabilities to their fullest. Each mode offers you its own unique advantages. Line mode is similar to EDIT/3000. You type in lines in response to a ">" prompt. Typing errors can be corrected by backspacing, which erases the errors and allows you to retype the line as it should appear. When you press the carriage return you send the line to RISE for execution.

Block Mode resembles VPLUS. You enter anything you wish on the screen in unprotected data fields. You may correct typing errors or alter the information displayed as needed. You position the cursor, using the cursor control keys and editing keys to make your changes directly on the screen. Afterwards, you press the ENTER key to send all the data on the screen to RISE for processing.

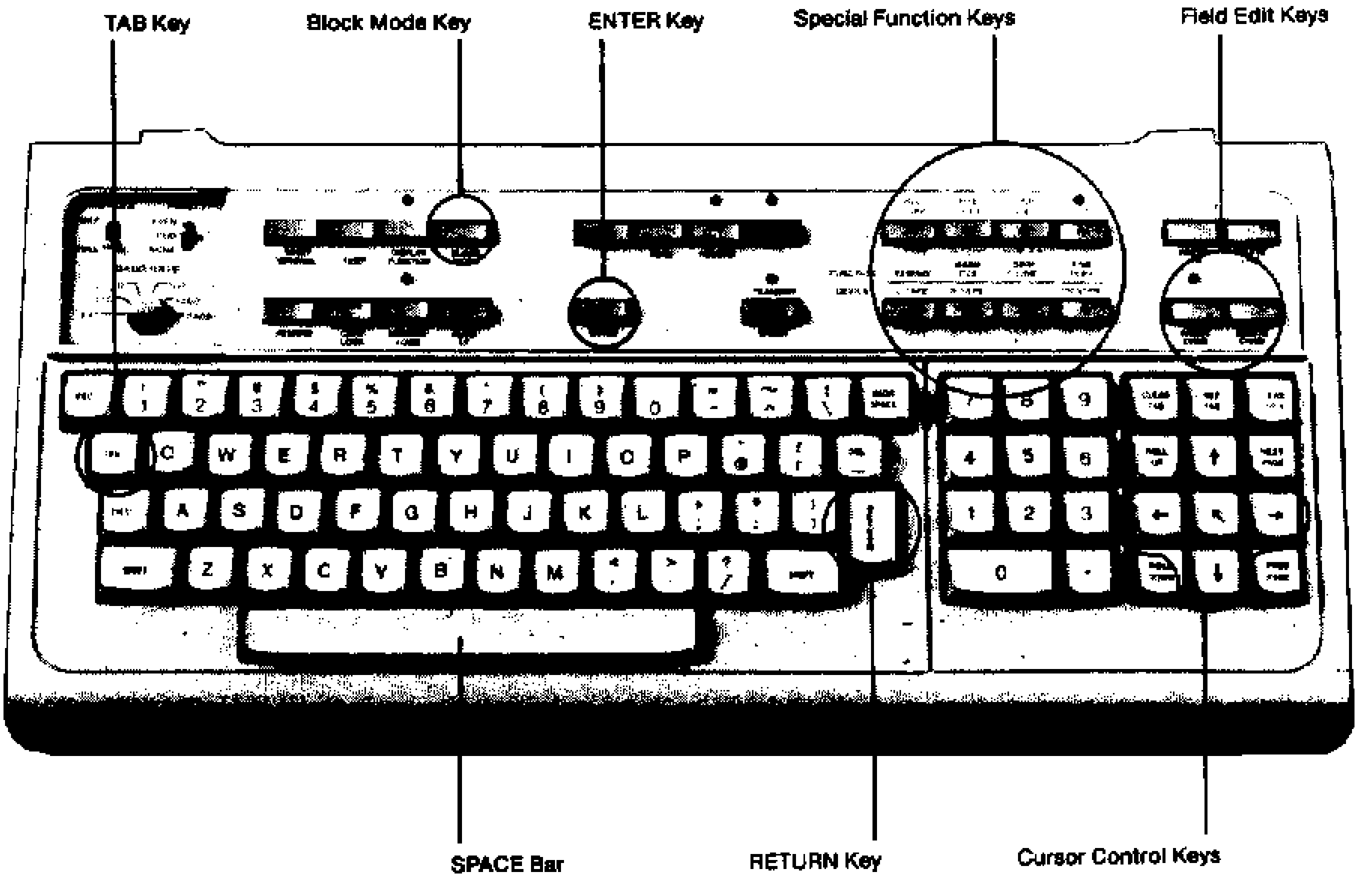


Figure 1-3. HP 2645 Terminal Keyboard

Line Mode Advantages

RISE is primarily a line mode editor. Line mode is effective because of the speed with which commands, one at a time, are sent to RISE for execution. Also, the effects of a command's execution are shown to you on the screen so that you see exactly what has occurred. For example, when you give the DELETE command and press the carriage return, RISE will display the lines being deleted.

Another advantage of Line Mode is its use of the terminal's display memory. Using the LIST command you can fill the screen and memory with text, then scroll up or down, or call the next or previous page for review, by pressing the terminal's special keys.

Block Mode Advantages

In Block Mode, you can make numerous changes to the data on the display and send them to RISE by pressing the ENTER key once. Your special function keys (softkeys) are activated to give you what is, effectively, pushbutton control. The purpose of the special function keys varies with Block Mode commands you have entered. Descriptive softkey labels appear at the top of the screen with each change of key function.

Commands Restricted to Modes

Certain commands operate only in Line Mode; others are restricted to Block Mode. Some work in either mode. If you give a command pertaining to one mode while in the other, RISE will automatically shift to the correct mode for that command. For example, if you enter the SHOW command while in Line Mode, RISE immediately shifts to Block Mode, and a display of RPG source lines will appear on your screen, allowing you to make any changes directly. Thus, all commands may be entered while in either mode.

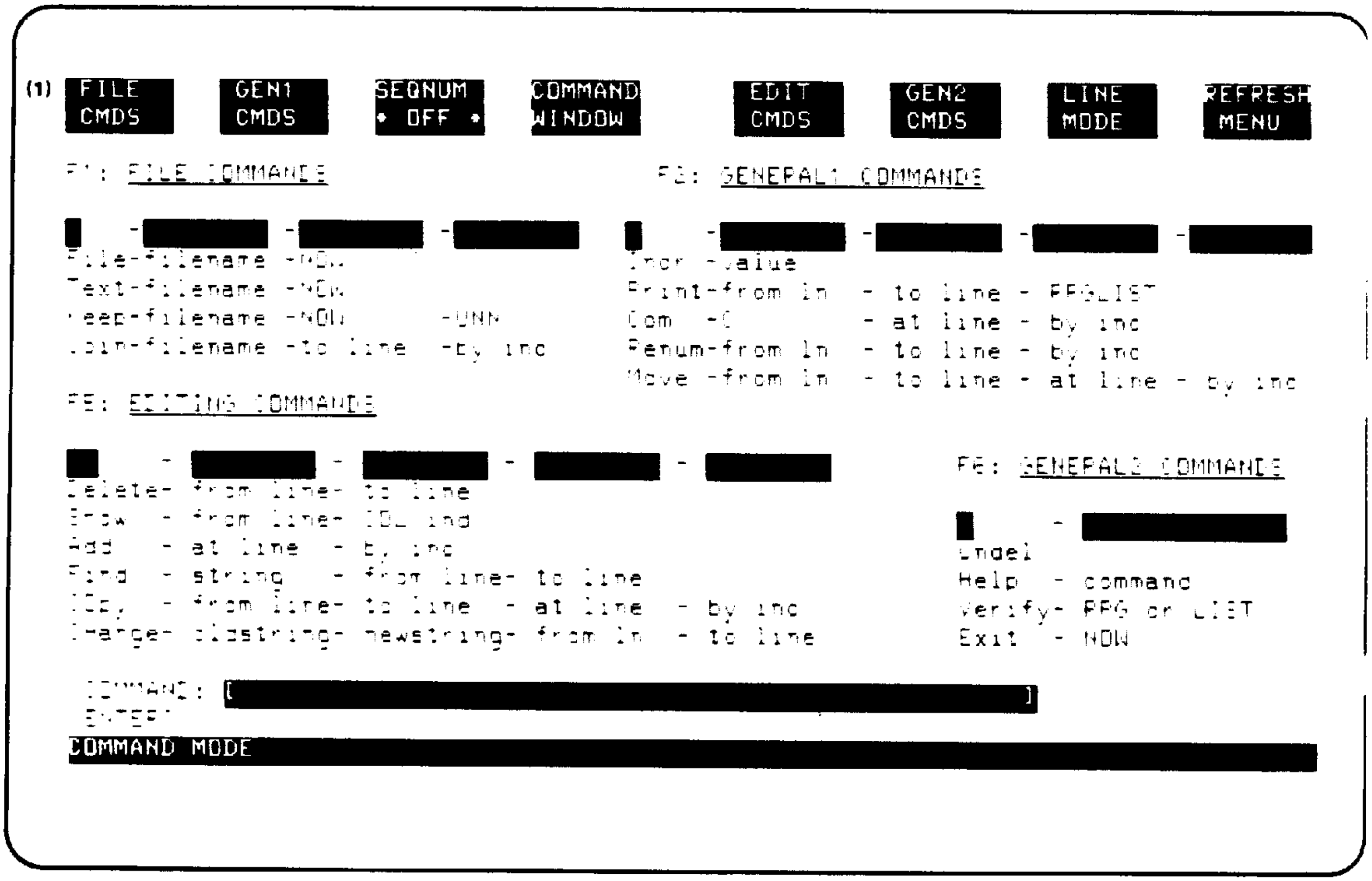
Summary of Modes

	Line Mode	**Block Mode**
Reached by	Special Function Key and automatic shift when you enter line command.	Automatic shift when you enter Block Mode command.
Commands entered by	Carriage return.	ENTER Key.
Advantages	Line-by-line editing. Commands executed with greater speed. Rapid, convenient editing methods.	Full screen display with RPG forms. Full screen editing with single entering procedure. Special Function Keys for push-button control.

Since RISE's command set is similar to that of EDIT/3000, familiarity with the editor will be of value in using RISE. It is not necessary, however. (See appendix A for a comparison of RISE and EDIT/3000.)

Features of Block Mode Command Menu

RISE offers you a command menu which lists the most important RISE commands so that you can select the next command you need. The commands are grouped by their functions: File Commands, Editing Commands, General-1 Commands, and General-2 Commands. You move the cursor to the group using a special function key labeled with the group title. Windows are provided for the parameters of each command. The menu relieves you of the need to memorize or look up the commands since it displays the options for a command. The menu operates only in Block Mode.



```
F1: FILE COMMANDS                             F2: GENERAL1 COMMANDS

File-filename -NOW                            Incr -value
Text-filename -NOW                            Print-from ln  - to line - RPGLIST
Keep-filename -NOW      -UNN                  Com  -C        - at line - by inc
Join-filename -to line  -by inc               Renum-from ln  - to line - by inc
                                              Move -from ln  - to line - at line - by inc
F5: EDITING COMMANDS

                                                           F6: GENERAL2 COMMANDS
Delete- from line- to line
Show  - from line- IDL ind
Add   - at line  - by inc                                  Undel
Find  - string   - from line- to line                      Help   - command
Copy  - from line- to line  - at line  - by inc            Verify- RPG or LIST
Change- oldstring- newstring- from ln  - to line           Exit   - NOW

 COMMAND: [                                              ]
 ENTER?
COMMAND MODE
```

Command Window

Besides the Command Menu, you have another means of entering commands while in Block Mode. You can use the Command Window which is always displayed on the terminal when you are in this mode. The Command Window is useful to those who are familiar with the commands and have no need to go through the menu but want immediate action. In such an instance, you enter the command in the window as you would enter it in Line Mode, using a simple syntax which is described in chapter 2. A Block Mode enter prompt (ENTER?) appears to the bottom left of the Command Window. This tells you that RISE is waiting for your next command. The enter prompt disappears when a command is being executed and returns upon completion.

Show Mode

The SHOW command will display an RPG Record Specification form at the top of the screen, followed by a page (10 lines) of RPG source records in Block Mode. If you have changes you wish to make to the records, move the cursor to those places where the changes are to be made. Pre-set tabs allow you to use the Tab key to skip quickly across the source records to important columns. When editing, you directly enter your changes on the screen so that what is visible duplicates exactly what will be stored in the file. When your editing is finished, you press the ENTER key to send your updated page to RISE. If, at that time, you want to continue with the next page of the source file, you push the special function key, F5 (Scroll forward). If you wanted to go into Line Mode, you would push F7. The Command Window can be used with displayed pages in Show Mode, enabling you to perform several editorial functions at the same time. For example, you may wish to modify the first page of text while deleting lines 100/150. To do this you would do the following: * SHOW the first page (10 lines) of the source code for the modification. (Your Command Window will also appear on the screen.) * Type in your modifications to the program directly. * Type in DELETE 100/150 in the Command Window. * Press the ENTER key. After you press ENTER, RISE reads in your changes, then pauses and executes your DELETE command, leaving you in the page modification mode of the SHOW command.

Special Function Keys (Softkeys)

The terminal special keys are fully integrated into the functional features of RISE. The editing keys and cursor positioning keys allow you to edit directly a displayed page. The special function keys give you push-button control of scrolling, mode shift, and a display of editor sequence numbers, among other capabilities. The function of the keys changes, depending upon the command which has been entered.

You do not have to memorize the function of the softkeys; descriptive labels associated with the activated keys appear at the top of the screen whenever the keys are available for use.

In the commands section of the manual (See chapter 2), the special function keys, also called softkeys, are discussed with the commands that bring them to the screen.

Files

The Work File

RISE performs all of your editing commands on a work file. The File and Text commands tell RISE which permanent file you wish to become your work file. The work file is a KSAM file (Keyed Sequential Access Method), and is used so that RISE can quickly random access your source records. (See the *KSAM/3000 Reference Manual* for more information on this file organization.)

The name of the work file is in the form Wdddhhmm, where "ddd" is the day of the year, "hh" is the hour of the day, and "mm" is the minute of the hour. The name of the associated key file is WKddhhmm, where "K" overlays the first "d" in the day of the year, and "ddhhmm" have the same meaning as they do with the work file. If there is an existing file with the same name, RISE will increment the last digit.

RISE automatically supplies sequence numbers to the KSAM work file in columns 81-88 if none already exist. These sequence numbers serve as search keys when RISE accesses the KSAM work file. In your command parameters you specify which source lines you want to work on. For example "LIST 1/10" will list to the screen the source lines whose seqence numbers are included in the range 1 to 10.

The renumber command allows you to manage the sequence numbers. Note that the sequence numbers are not associated with the RPG line numbers in columns 1-5 of your source program. The sequence numbers are not a part of the RPG program.

There are several ways in which you may specify which sequence numbers you want processed. You may give the precise number, as in "LIST 1/10", or you may use relative numbers in relation to the current position of the record pointer. You may also use the literal specifications FIRST, LAST, or ALL. More information concerning the specification of sequence numbers with various commands appears in the command section (chapter 2).

Accessing Files

When you TEXT your files, your permanent file is copied into a work file (which is a KSAM file). It is copied back to a permanent file when you use the KEEP command. Using TEXT and KEEP you edit work files only, bringing them to the screen for editing and storing them as permanent files when the work is concluded.

The TEXT/KEEP commands provide a safeguard in that the editing done to a work file is not made permanent until the KEEP is given. Thus, if erroneous deletions were made to the work file, you would retain the deleted lines in the permanent file. Using the TEXT command you could copy it back to the work file in its original state for further work at the terminal.

The price of the safeguard is additional time required to copy the files back and forth between your permanent file and the RISE work file. If your RPG source file is long, the overhead may be considerable.

You can also edit your files directly using the FILE/EXIT commands. Only one copy of the file is created with this option. RISE edits the copy directly so that no time is wasted.

In using FILE/EXIT you sacrifice your safeguard, however. You do not have a permanent copy which can be recalled if, for example, you wish to restore lines mistakenly deleted. Changes made to a file called with the FILE command cannot be rescinded unless you made a backup copy before editing it.

In general, FILE/EXIT serve best when you are adding to or editing a large RPG source file. When your file is small, TEXT/KEEP are recommended. Also, for a more efficient use of space, TEXT/KEEP are recommended for large files that are debugged, complete, and in production. Such files rarely need additional editing. Storage of a regular sequential file takes less disk space than storage of the same file in KSAM format because of the extra space consumed storing the associated key file.

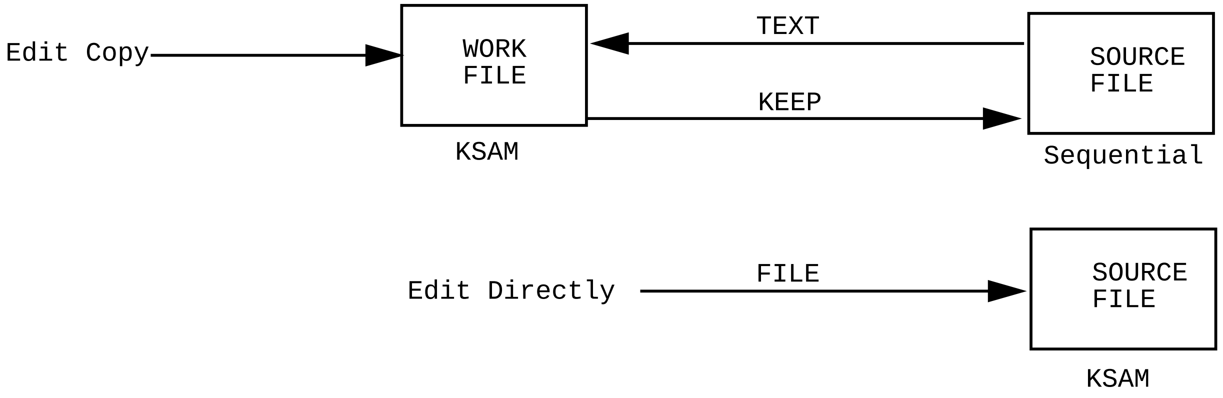


Figure 1-4. RISE Editing Methods

Safeguards

RISE has two important built-in safeguards which help to protect you from two serious errors that you may accidentally make. The first safeguard is associated with the TEXT, FILE, EXIT, and BEGIN commands. These commands always clear the work file, and there is a danger that you might clear or destroy a work file you have been editing because you entered one of these commands without first keeping your recent changes to a permanent file. When there is a possibility that this might happen, RISE will prompt you with the following question:

```
        "KEEP not done.  CLEAR current work file?"
```

Thus you are reminded that the changes you have made while editing will not be saved if your TEXT, FILE, EXIT, or BEGIN is immediately executed. You are given the opportunity to decide whether or not you wish to save the changes first. You may respond "NO" to the question if you want to cancel the command that you entered; or you may respond "YES" if you do want RISE to go ahead and clear the work file, executing your command without saving your changes.

The second safeguard is associated with the KEEP command. It protects you from overwriting an existing permanent file. If you try to KEEP the contents of your work file to a name of a file that already exists, RISE will prompt you as follows:

```
"File exists already.  Purge old 'filename'?"
```

This gives you the opportunity to decide whether or not you want to overwrite the pre-existing permanent file. If you do, you respond "YES" and RISE will execute your KEEP command. If you do not want to lose the contents of the pre-existing file, you respond "NO" and RISE will cancel the execution so that you can KEEP to a different file name if you wish.

There will be occasions when you have decided beforehand that immediate execution of the TEXT, FILE, EXIT, BEGIN, and KEEP commands is precisely what you want. In such an instance, the prompts would cause unnecessary delays. You have an optional parameter called "NOW" which saves time by disabling RISE's safeguards. If you specify "NOW" with any of these five commands, RISE will execute immediately without prompting you to reconsider. You use the parameter when you are certain of the results you want.

Running RISE

To run RISE you enter the following MPE command at your terminal:

```
RUN RISE.PUB.SYS
```

Your command will be answered with an HP banner like the one below:

```
HP32104A.05.00 RPG INTERACTIVE SYSTEM ENVIROHMENT  RISE
(C) HEWLETT-PACKARD CO. 1981     WED, SEP 16, 1981, 10:29 AM
>
```

With that you have begun the program and entered the RPG Interactive System Environment. Other MPE commands are available for your use within the system. A list of these commands may be obtained by typing "HELP:".

The commands are displayed on the screen in Block Mode.

To leave the system you use the EXIT command.

Because of the visual and interactive emphasis in the design of RISE, it is not recommended that you use it in batch mode.

RISETOUR--A Self-paced Guided Tour of RISE

RISETOUR is a self-paced introduction to RISE. It is on-line and can be called by the following commands:

```
:RUN RISE.PUB.SYS

HP32104A.05.00 RPG INTERACTIVE SYSTEM ENVIRONMENT  RISE
(C) HEWLETT-PACKARD CO. 1981     WED, SEP 16, 1981, 10:29 AM

>

>GET RISETOUR.PUB.SYS
```

However, before you initiate RISETOUR, you must have the file SIMCAL in your group and account. SIMCAL is an RPG source program referenced in RISETOUR (and used extensively in the examples included in this manual). To get a copy of SIMCAL into your group and account, you can use FCOPY to copy it from "SIMCAL.PUB.SYS" to "SIMCAL.yourgrp.youracct".

RISETOUR is interactive; it contains many requests and directions which call for your direct participation in the tour. It is automatic and controls the complete tour. Using it, you will quickly develop a familiarity with most of RISE's features first-hand.

The tour is divided into lessons, and the lessons are grouped under the following headings:

* File manipulation commands.
* General editing commands.
* Block mode.
* RPG dedicated commands.
* Calling the RPG compiler.

RISETOUR is designed as an introduction rather than an intensive training course. Messages are provided to explain the relationship between what appears on the screen and its application to the needs of an RPG programmer. Pauses are imbedded to allow time for you to study the examples on the screen.

RISETOUR can also be useful as a printed reference guide. If you text in "RISET OUR.PUB.SYS" and use the "LIST ALL OFF" command, you will get a listing of the tour offline on a line printer. This will allow you to review at length parts of the tour without the delays caused by scrolling through the screen version.

RISE Commands

RISE only accepts commands that are typed in capital letters. This restriction will prevent you from accidentally entering your RPG source code in lower case letters, which the RPG Compiler will not accept.

These commands are listed below with brief explanations of their use. The portions of the commands outside the brackets are the minimum abbreviations that RISE will recognize and accept.

Futhermore, any number of letters in the optional portion of a command enclosed within brackets may be given and will be recognized by RISE. For example, in the "E[XIT]" command, the "E" is required, and the "XIT" portion is optional. The following variations are all acceptable:

```
E
EX
EXI
EXIT
```

Commands List

```
1.  A[DD]            --add new lines

2.  B[EGIN]          --start editing a new work file

3.  CH[ANGE]         --change oldstring to newstring

4.  COM[MENT]        --create a comment banner for user documentation

5.  COP[Y]           --duplicate lines

6.  D[ELETE]         --delete lines
```

7.	E[XIT]	--end subsystem
8.	FIL[E]	--work file directly
9.	F[IND]	--locate a string
10.	FO[RM]	--display an RPG specification form on the screen
11.	G[ET]	--read and execute commands from a file
12.	H[ELP]	--explain commands
13.	I[NCR]	--set default increment value
14.	J[OIN]	--append or merge an entire file
15.	K[EEP]	--save the work file
16.	LIN[E]	--enter Line Mode
17.	L[IST]	--list lines in Line Mode
18.	ME[NU]	--display main menu
19.	M[ODIFY]	--modify lines
20.	MOV[E]	--move lines to new location
21.	P[RINT]	--print lines offline
22.	R[ENUM]	--renumber editor sequence numbers or RPG source line numbers
23.	RU[N]	--run a program file
24.	S[HOW]	--display a page in Block Mode for direct screen editing
25.	T[EXT]	--copy a file into a work file for editing
26.	U[NDEL]	--restore lines deleted by last DELETE command
27.	V[ERIFY]	--compile the work file or LIST the compilation listing or prepare a USL file
28.	X[PAND]	--expand the work file size
29.	:MPEcommand	--execute MPE command

The RISE commands can be grouped together by the mode in which they execute and the functions they fulfill. It is not necessary to memorize the modes in which commands operate. If you give a command unsuited to the mode in which you are operating, RISE will automatically shift you to the mode that will accept and execute it. The HELP command provides additional convenience by listing and explaining RISE commands on the screen.

The following commands execute only in Line Mode:

```
        ADD                 BEGIN                FORM                 LINE
        LIST                MODIFY               RUN                 :MPEcommand
```

The following commands execute only in Block Mode:

```
        COMMENT          HELP             MENU             SHOW         VERIFY
```

The following commands execute in either mode:

```
        CHANGE              COPY                DELETE              EXIT
        FILE                FIND                GET                 INCR
        JOIN                KEEP                MOVE                PRINT
        RENUM               TEXT                UNDEL               XPAND
```

The following commands, if entered in Block Mode, will automatically return you to the main menu when the command is completed:

```
        FILE                          RENUM                            TEXT
```

These commands alter the internal state of the work file. What shows on the screen after they have been entered in Block Mode is not representative of what has occurred internally. The main menu resets the screen contents so that they are representative.

For example, suppose you are in Show mode with the first page of File1 displayed on the screen and find it necessary to edit File2. You would then enter the command "TEXT FILE2". After RISE completes the execution, your new internal work file will contain the contents of File2. However, the screen will still be displaying the first page of File1. At this point, RISE will automatically clear the screen by displaying the main menu. Then you may enter the Show command to have the first page of File2 displayed.

Commands Divided by Function

The following commands are grouped by function. The division presented here appears on the main menu. See MENU for a description of how you use the cursor and softkeys to execute the commands from the main menu. Note that not all commands appear on the main menu, but all the more important ones do.

```
F1: FILE COMMANDS                  F2: GENERAL1 COMMANDS
        FILE                                   INCRement
        TEXT                                   PRINT
        KEEP                                   COMment
        JOIN                                   RENUMber
                                               MOVE

F5: EDITING COMMANDS               F6: GENERAL2 COMMANDS
        DELETE                                 UNDELete
        SHOW                                   HELP
        ADD                                    VERIFY
        FIND                                   EXIT
        COPY
        CHANGE
```

Command Limitations

RISE will not accept commands longer than 50 characters. This limitation is established by the number of characters which will fit into the Command Window. It holds true with all commands, whatever the mode in which they are entered.

Also, multiple commands entered on a single line are not accepted by RISE.

RISE does not accept special characters used to separate parameters for a command. This is a convenience designed to relieve you of the need to remember which special characters to use for delimiters. For example, "LIST ALL OFF" is correct; "LIST ALL, OFF" is not because of the special character (the comma).

It is necessary, however, to use the space bar to separate the parameters with one or more blank spaces except when the parameters are of a different type (alphabetic vs. numeric). For instance, in the command "FIND ‘string' IN FIRST", the space separating IN and FIRST is required since both parameters are alphabetic. If there was no space, RISE would not be able to recognize the parameters. However, the command "FIND ‘string' IN1", would be accepted without a space separating IN and 1 because the parameters are of different types, and RISE readily distinguishes between the two.

For your convenience, RISE will accept the F[ORM] and COM[MENT] commands without a space between the command and the alphabetic parameter. Whenever RISE encounters an unacceptable use of parameters, an error message appears on the screen to explain the problem.

10 COMMANDS-DETAILED DESCRIPTIONS

Notations for this Manual

Notation	Meaning	Example of Use
[opt]	Part of command or parameter enclosed by brackets is optional.	A[DD] [^^[formf]]
form	One of the designators of the RPG Record Specifications--H,F E,L,I,C,O--or CO[L], which brings a column indicator to the screen.	FO[RM] form
lb	Line begin (beginning with line).	D[ELETE] lb
le	Line end (ending with line).	D[ELETE] lb/le
inc	Increment value.	A[DD] BY inc
/	Delimiter with semantics of TO.	L[IST] lb/le
X		A[LL]
	Choose one of the options, X or Y.	L[IST]
Y		lb/le

Literal Parameters

Notation	Meaning	Example of Use
BY	The following number is an increment value.	A[DD] BY.01
CO[L]	Display with column indicator.	FO[RM] CO[L]
TO	The following line number is a destination; or, the following is a new string.	COP[Y]l TO l0 CH[ANGE] "OLD" TO "NEW"
IN	The following is a range of line numbers.	F[IND] "string" IN lb
E[VERY]	Locates every occurrence of string in the FIND command.	F[IND] E[VERY] "string" IN A[LL]
O[FF]	List lines offline on a line printer in the LIST command.	L[IST] A[LL] O[FF]
N[OW]	The command is to be executed immediately without safeguards.	E[XIT] N[OW]

U[NN] U[NN]	The work file is to be made permanent without the appended sequence numbers.	K[EEP] SAVEFILE
A[LL]	The command applies to the entire work file.	D[ELETE] A[LL]
F[IRST]	Indicates first line in work file.	COM[MENT] F[IRST]
L[AST]	Indicates last line in work file.	M[ODIFY] L[AST]
^^	Controls the display of the RPG Record Specification form.	A[DD] ^^H
R[EPEAT]	Controls the auto-repeating factor in ADD mode.	A[DD] ^^ HR[EPEAT] 3
*	Indicates current position of record pointer (in Line Mode only).	L[IST] *-5/*+5
@	May be used in place of *.	L[IST] @-5/@+5

Those command descriptions which include references to <Control Y> and the use of "*" for the record pointer position apply only if RISE is in Line Mode.

Specifying the Sequence Numbers

There are two ways of specifying sequence numbers: you can use references to actual line numbers, or you can refer to them relative to the current position of the record pointer. The record pointer is located at your current editing sequence number in the file. In this instance, only the pointer's position relative to sequence numbers is used.

Below is a summary of the ways in which you may specify sequence numbers using RISE:

Actual specification	L20/37 (In the manual notation, this will be written as "lb/le", meaning line beginning and line ending.)
Actual specification	F[IRST]/L[AST] may be used in place of lb/le
Actual specification	A[LL] may be used in place of lb/le
Relative specification (The special character, *, indicates the current	L* -5/* +5

position of the record pointer.)	
(The special character, @, may be used in place of * to indicate record pointer position.)	L@-5/@+5
(Specifications may be mixed.)	LF[IRST]/* +10 L* -5/L L 10/L

Increments for Sequence Numbers

Every text record in RISE's work file has a unique sequence number that RISE uses to access the KSAM work file. The difference in value between each sequence number is known as the increment. RISE uses a default increment value of 1 which you may replace with the INCR[ment] command followed by the new value, so long as that value is between .001 and 5000 inclusively.

Those RISE commands which create new sequence numbers and text records for you need an increment value to determine the next new number. Some of the commands involved are RENUM, ADD, COPY, MOVE, and JOIN. Each of the commands provides the optional paramater "BY inc" to allow you to specify the increment for RISE to use while executing the command. If you omit "BY inc", RISE will use the current default increment value.

At times you will want to add or merge new text records between two existing sequence numbers. If the current increment value that RISE is using is too large, the new sequence numbers being generated may exceed the existing upper sequence number. If this happens, RISE will automatically make room for your new lines by reducing the large increment to a smaller one to stay within the upper limit while keeping your text records in proper order.

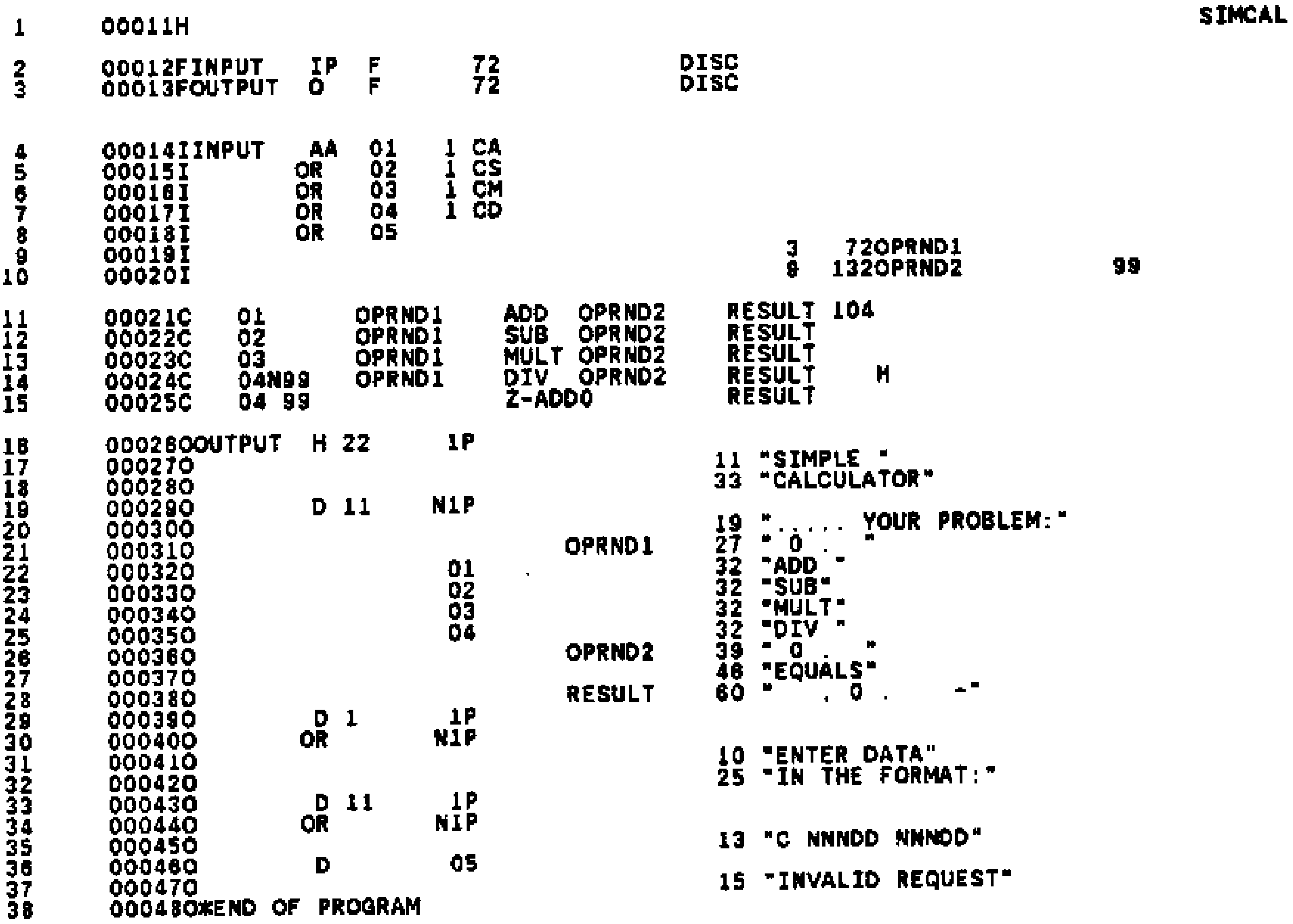
If, however, RISE cannot determine a small enough increment value because you are attempting to introduce too many lines into a restricted space between two existing line numbers, RISE will display the following error message:

```
     Error 144: No room between line numbers.
```

In such a case, you can use the RENUM[ber] command to change the sequence numbers and continue with your editing.

About the Examples

Examples of commands in the section which follows use a short RPG source file called SIMCAL (SIMple CALculator). The program will perform simple calculations--addition, subtraction, multiplication, and division. Here is a listing of the file which was made using RISE's PRINT command.



```
                                                                                          SIMCAL
00011H
00012FINPUT    IP  F      72            DISC
00013FOUTPUT   O   F      72            DISC

00014IINPUT    AA  01    1 CA
00015I        OR   02    1 CS
00016I        OR   03    1 CM
00017I        OR   04    1 CD
00018I        OR   05
00019I                                          3   72OPRND1
00020I                                          8  132OPRND2           99

00021C   01      OPRND1    ADD  OPRND2    RESULT 104
00022C   02      OPRND1    SUB  OPRND2    RESULT
00023C   03      OPRND1    MULT OPRND2    RESULT
00024C   04N99   OPRND1    DIV  OPRND2    RESULT   H
00025C   04 99             Z-ADD0         RESULT

00026OOUTPUT   H 22     1P
000270                                    11 "SIMPLE "
000280                                    33 "CALCULATOR"
000290         D 11   N1P
000300                                    19 "..... YOUR PROBLEM:"
000310                          OPRND1    27 " 0 .  "
000320                  01                32 "ADD "
000330                  02                32 "SUB"
000340                  03                32 "MULT"
000350                  04                32 "DIV "
000360                          OPRND2    39 " 0 .  "
000370                                    46 "EQUALS"
000380                          RESULT    60 "   , 0 .    -"
000390         D 1      1P
000400        OR        N1P
000410                                    10 "ENTER DATA"
000420                                    25 "IN THE FORMAT:"
000430         D 11     1P
000440        OR        N1P
000450                                    13 "C NNNDD NNNDD"
000460         D        05
000470                                    15 "INVALID REQUEST"
00048O*END OF PROGRAM
```

To compile, prepare, and run SIMCAL from within the RPG Interactive System Environment, you would enter the commands below. The comments on the right are for your information and are not part of your responses. While the Segmenter is executing, its HP banner and responses will appear on your screen. You are expected to supply colon (":") prompts as required when entering the ":MPEcommand".

```
------------------------------------------------------------------------------------

>T[EXT] SIMCAL                              Copy SIMCAL to work file

>V[ERIFY] R[PG]                             RPG compile the work file into a  USL file

>PRINT ALL RPGLIST                          Prints compilation listing  offline

>V[ERIFY] P[REP]                            Prepare USL to program file

>:FILE INPUT=$STDIN                         Issue file equations for I/O
>:FILE OUTPUT=$STDLIST

>RU[N] $OLDPASS                             Execute SIMCAL
```

Alternatively, you could prepare SIMCAL manually using the commands listed below. Again, you will have to supply colon prompts where they appear below. These are examples of the ":MPEcommand".

(See:MPEcommand.)

RISE will respond with execution messages, mode changes, and banners to some of the commands. The example below includes only the prompt for information to be used in running the calculator. Comments appear to the right and are not part of your responses. Also, see the *MPE Segmenter Reference Manual*, for more information on USL and program files.

```
---------------------------------------------------------------------------------------

>T[EXT] SIMCAL                              Copy SIMCAL to work file

>L[IST] A[LL]                               List contents for review

>V[ERIFY] R[PG]                             RPG compile the work file

>PRINT ALL RPGLIST                          Prints compilation listing offline

>RU[N] SEGDVR.PUB.SYS                       Run the Segmenter subsystem

-USL $OLDPASS                               Identify the USL file

-PREPARE $NEWPASS                           Prepare USL file in a new program file

-EXIT                                       Leave the Segmenter subsystem

>:FILE INPUT=$STDIN                         Issue FILE equations for I/O

>:FILE OUTPUT-$STDLIST

>RU[N] $OLDPASS                             Execute SIMCAL
```

When SIMCAL executes, you will receive the following message:

```
     SIMPLE     CALCULATOR

     Enter the data in the format:
     C     NNNDD        NNNDD
```

Under C, you enter S (subtract), M (multiply), D (divid e), or A (add). Your operands may include three whole numbers (NNN) and two decimals (DD).

After you enter the data in the prescribed format, SIMCAL produces the arithmetic answer and prompts for more data.

When you have all the calculations you require, type ":" (colon), and you will exit from SIMCAL and return to RISE. Do not type ":EOD" to end SIMCAL for it will also end RISE.

ADD

ADD allows you to add new lines to a work file.

Form

```
     A[DD] [lb] [BY inc] [^^[form] [R[EPEAT] [n]]]
```

Parameter Explanation

lb The line number where you wish to start adding lines. If not specified, adding begins after the last line in the work file. If "lb" exists in the work file, RISE will first display the existing source record at "lb" and prompt you with a sequence number immediately following "lb".

BY inc Increment by which you wish to add lines. If omitted the current default is used.

^^ Your control of the RPG Rec ord Specification forms. By itself "^^" homes the cursor and clears the screen before you begin adding lines. The "^^" option and its parameters may be entered while in ADD mode at the start of a new line to change the RPG Record Specification form or the REPEAT option. (See R [n] below.)

form Additionally displays the RPG Record Specification form type you designate. "form" may be H,F,E,L,I,C,O, or CO[L].

Rn The REPEAT command homes the cursor and clears the screen of all but the Record Specification form or column indicator after your have entered "n" number of lines. If "n" is omitted or "n" equals zero, the auto repeat is turned off.
Note: REPEAT parameter must be preceded by "^^".

Purpose

The ADD command allows you to add new lines to a work file. If there is no work file established--if you have not entered a TEXT or FILE command--your ADD command will automatically create a new work file.

Related Commands

While in ADD mode, you may enter the LIST MODIFY, and DELETE commands. To do so, you type ">>" followed by the command at the start of a new line.

If you type ">>" without a command, the cursor moves to column 6 of your RPG Record Specification form. This relieves you of the tedious task of spacing over columns 1 through 5, where the RPG optional line numbers are specified, in order to begin entering your record specifications.

Once you type ">>" alone at the start of a new line, RISE will always prompt you for new text starting from Column 6 whenever you are in ADD mode.

To return the cursor to column 1, type "<<", and RISE will prompt you from there.

Leaving ADD

To leave the ADD mode you type <Control Y> or "//" at the start of a new line.

Execution mode: Line

Record pointer: At last added line

Examples

The following are legal abbreviations used with the ADD command and its parameters:

A10^^H R3 Add line 10, Header Specifications form, REPEAT 3.

A^^H Add under Header Specification form.

AD L BY 10 Add beginning after last line using an increment value of 10.

Examples in this manual are based upon the file SIMCAL. In the examples of the ADD command, a portion of SIMCAL is entered. Various parameters are used. Errors are committed, and the messages appear with the listing.

In the first example below, when the ADD command creates a work file, the ADD begins at line 1 because the file is empty. If the file existed previously, RISE would display the last line in the file and the next line number, where the ADD would begin.

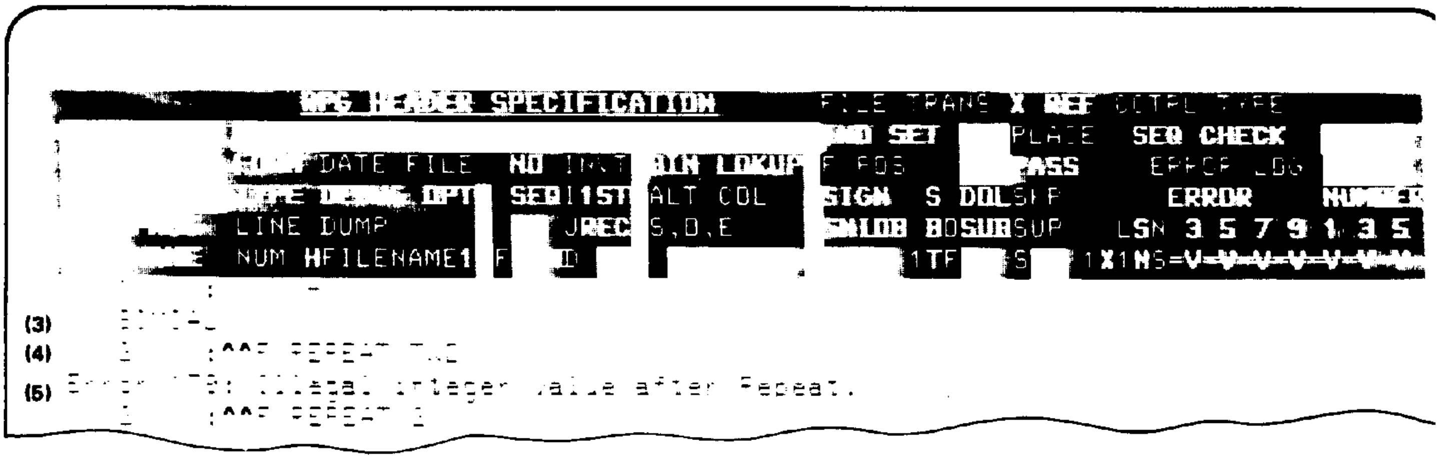
RISE prompts with line numbers for you to enter text. Two slashes (//) at the beginning of a line terminate the ADD command. The ADD command can also be terminated by <Control Y>.

Initially, in the first example, an error is committed when the ADD command is given in lower case letters. RISE only accepts capital letters (1). Following the error, the ADD command is legally entered with a message and line number prompt (2). No text is entered, however. Instead, a command is entered which calls for a display of the column indicator (3). This command could have been entered immediately following the ADD command in the following form:

```
     A[DD] ^^COL

     (1) >add
         Error 195: Only capital letters are allowed.

     (2) >ADD

     (3) Creating work space file.
           1:   ^^COL
```

In the next example, in which the column indicator is displayed, an error is committed when an illegal Record Specification form is commanded (1). Following the correct entry of a command for a Header Specification form (2), the form is displayed and the first line of text added (3). Another error is made when an alphabetic integer (TWO) is used with the R[EPEAT] parameter (4). Following the error message, the command is given correctly (5).

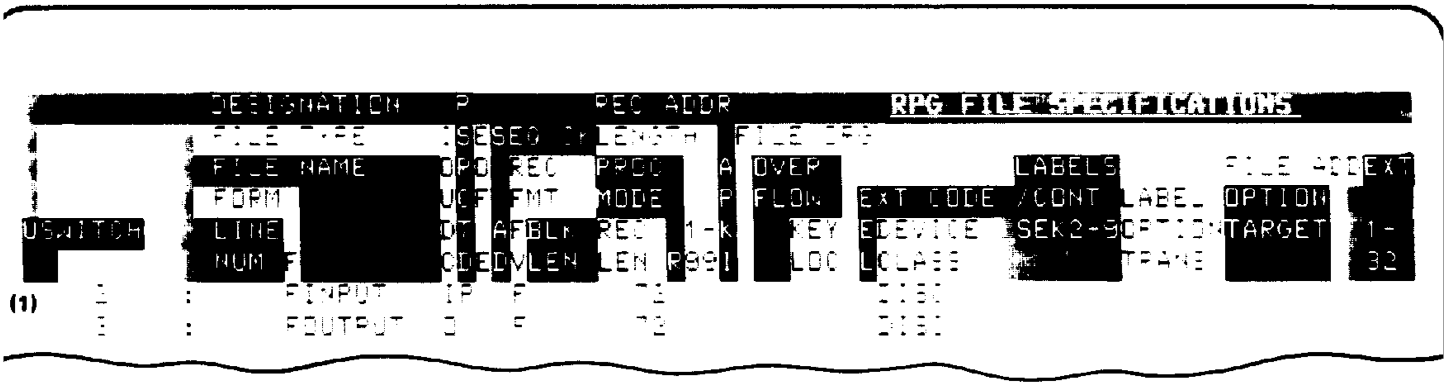
```
           0         1         2         3         4         5         6         7
     (1)
12345678901234567890123456789012345678901234567890123456789012345678901234567890
             1    :^^X

     (2)  Error 176:  Only H,F,E,L,I,C,O,COL or Repent allowed after ^^.
             1    :^^H
```



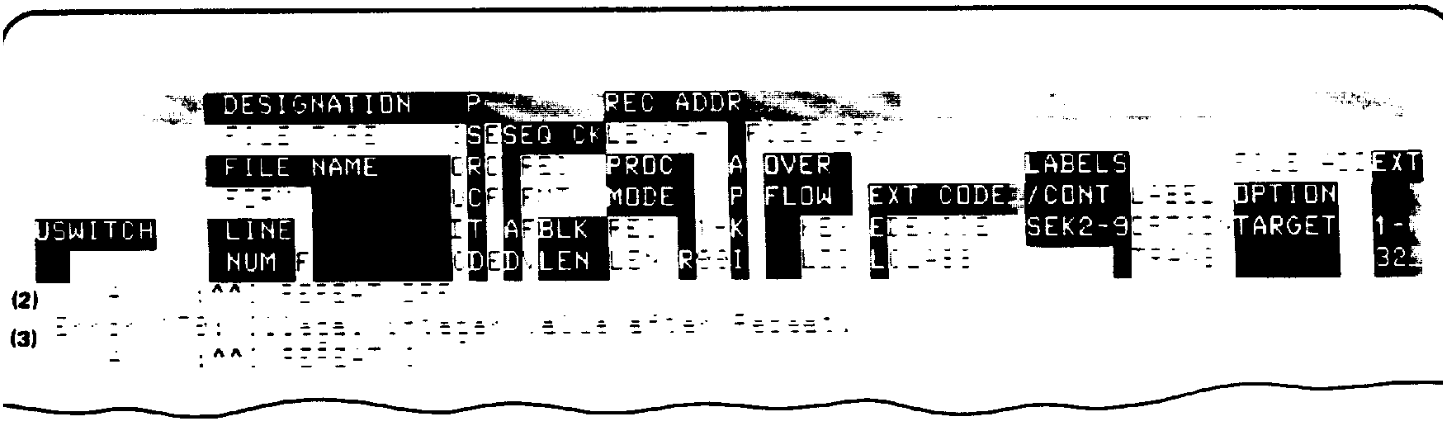
```
          RPG HEADER SPECIFICATION        FILE TRANS X REF CCTRL TYPE
                                           ND SET      PLACE   SEQ CHECK
        UPDATE FILE   NO INST  BIN LOKUP F POS          PASS    ERROR LOG
        DEBUG OPT     SEQ 1ST  ALT COL   SIGN   S DOLSIF     ERROR          NUMBER
         LINE DUMP       JREC  S,D,E     NLOB  BOSUBSUP    LSN 3 5 7 9 1 3 5
         NUM HFILENAME1 F   D                1TF     S    1X1NS=V-V-V-V-V-V-V
   :   -
(3)  EDITAL
(4)  4    ;^^R REPEAT TWO
(5) Error 178: Illegal integer value after Repeat.
     4    ;^^R REPEAT 2
```

In the next example, lines 2 and 3 are entered (1). With the ^^R[EPEAT]2 parameter in effect, the second screen display shows that lines 2 and 3 have cleared automatically, the cursor has homed, and line 4 has been added (2). The new line contains a syntax error in the attempt to shut off the REPEAT parameter while changing the specification form to I, for Input Specifications (3).

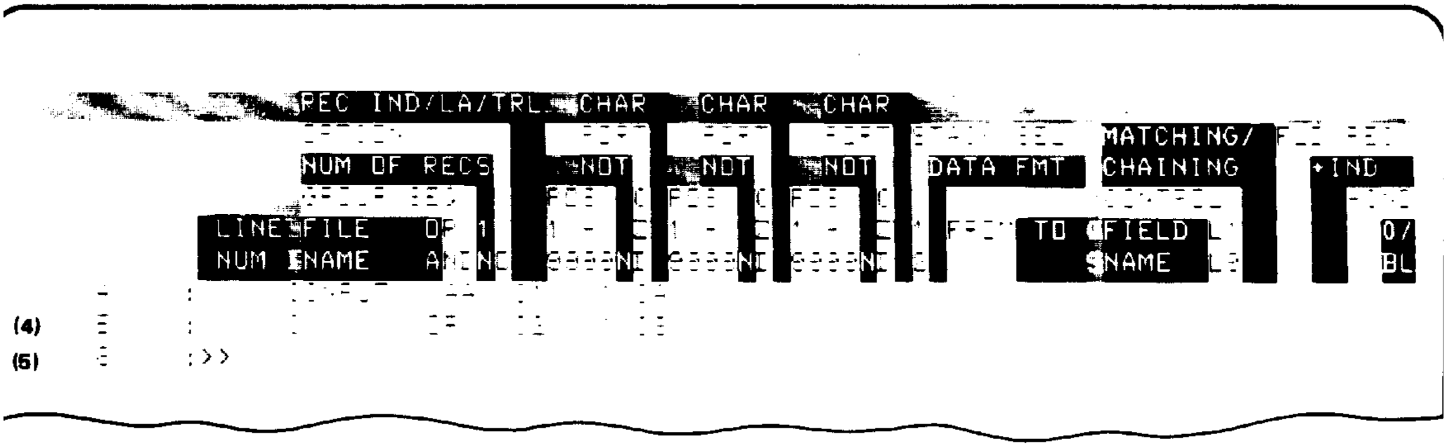
When the command is entered correctly, the Input Specifications form is displayed and lines 5 and 6 are added (4). Line 6 contains a command (">>") which instructs RISE to move the cursor to column 6 of the Record Specification form in order to make the entry of your specifications more convenient (5).



```
                DESIGNATION   P         REC ADDR               RPG FILE SPECIFICATIONS
                FILE TYPE     ISESEQ CKLENGTH   FILE ORG
                FILE NAME     OPO REC  PROC    A OVER                   LABELS        FILE ADDEXT
                FORM          UCF FMT  MODE    P FLOW  EXT CODE  /CONT  LABEL  OPTION
USWITCH         LINE          DT AFBLK REC  1-K   KEY  EDEVICE  SEK2-9OPTIONTARGET 1-
                NUM  F        CDEDVLEN LEN R99I   LOC  LCLASS          TRANS         32
(1)   1   :        FINPUT     IP  F        72          DISK
      2   :        FOUTPUT    O   F        72          DISK
```



```
                DESIGNATION   P         REC ADDR
                FILE TYPE     ISESEQ CKLENGTH   FILE ORG
                FILE NAME     CRC REC  PROC    A OVER                   LABELS        FILE ADDEXT
                FORM          LCF FMT  MODE    P FLOW  EXT CODE  /CONT  LABEL  OPTION
USWITCH         LINE          IT AFBLK REC  1-K   KEY  EDEVICE  SEK2-9OPTIONTARGET 1-
                NUM  F        CDEDVLEN LEN R99I   LOC  LCLASS          TRANS         32
(2)   9   :^^: [illegible]
(3) Error [illegible]: Illegal integer value after Repeat.
      9   :^^: [illegible]
```

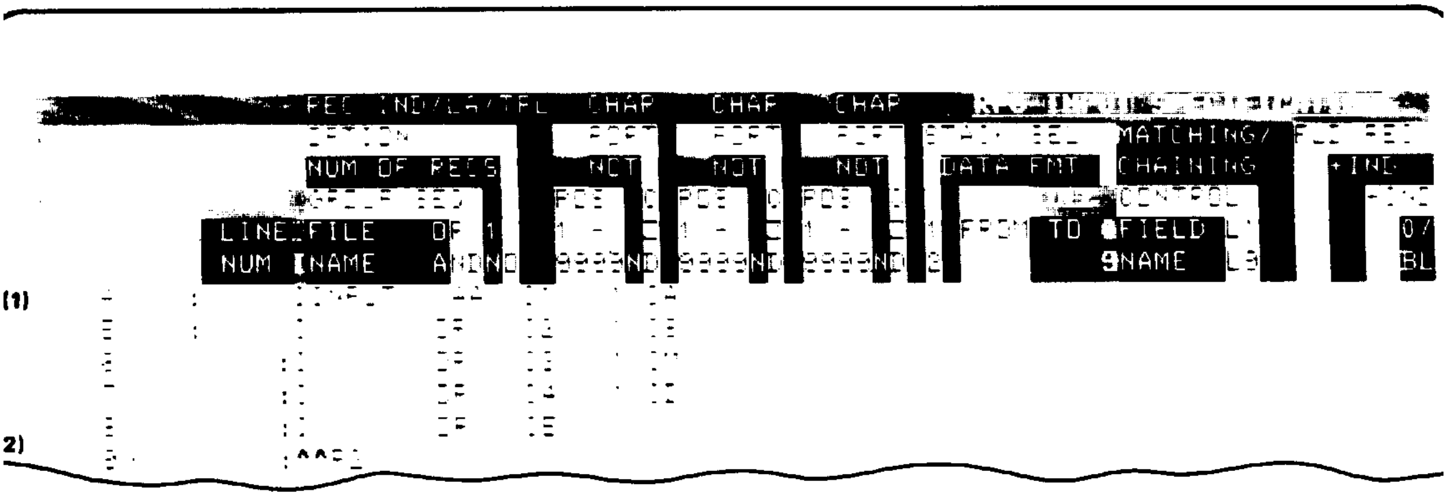


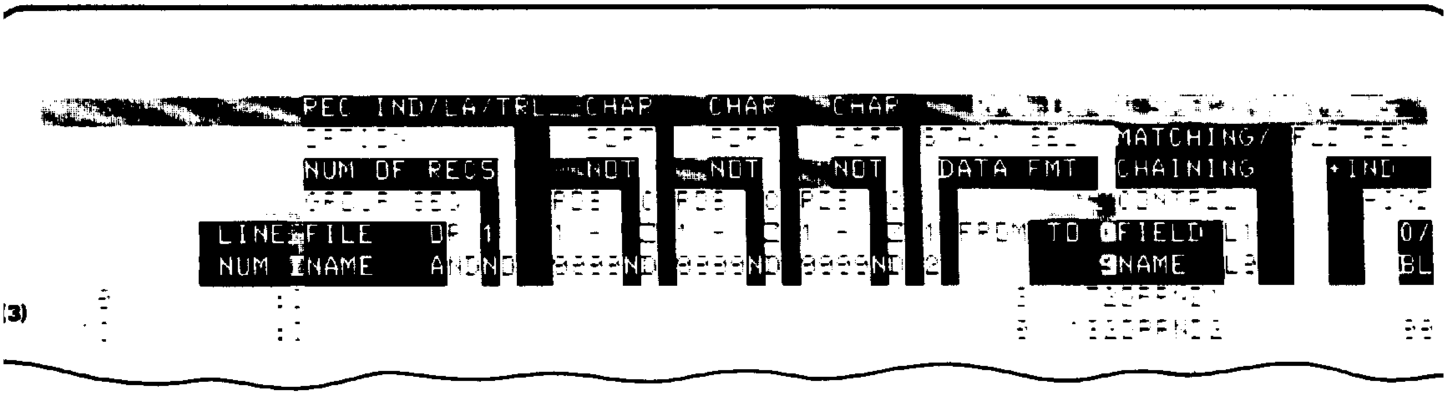
```
                REC IND/LA/TRL CHAR  CHAR  CHAR
                [illegible]                               MATCHING/ [illegible]
                NUM OF RECS     NOT   NOT   NOT   DATA FMT CHAINING   +IND
                [illegible]
          LINE FILE    OF 1   1 - C 1 - C 1 - C  [illegible] TO FIELD L    0/
          NUM  NAME    ANINC  9999NC 9999NC 9999NC          $NAME L    BL
      10  :  [illegible]
(4)   11  :  [illegible]
(5)   11  :>>
```

The remaining examples show lines added correctly with the prompt at column 6 (1). In line 9, the R[EPEAT] parameter is again in effect (2). The display shows the results after two Input Specification records are entered (3). After the second line is entered (line 10), the cursor will home up and prompt for more text with line 11.

Line 11 shows the way in which you may use the L[IST] command in ADD mode if it is preceded by ">>" (4). D[ELETE] and M[ODIFY] may also be used in this mode if preceded by ">>".

<Control Y> is entered at the start of line 11 to exit ADD mode. RISE prompts you for your next command with the ">" (5).





[ERROR:] Click here to view figure.

BEGIN

BEGIN allows you to initiate a new editing session as if you had just entered the RPG Interactive System Environment.

Form

```
B[EGIN] [N[OW]]
```

Parameter Explanation

N[OW] If you are editing a work file initiated with the TEXT command, NOW will cause the command to execute immediately, and it will disable the safety prompts. If omitted, and you have made changes to the work file, RISE will prompt with the following message:

"KEEP not done; CLEAR current work file?"

If you respond "YES", the command will execute, and the file will be purged

A "NO" will cancel the command.

Purpose

The BEGIN command achieves effects similar to entering an EXIT command, thus leaving RISE to end the current editing session, and then reentering RISE to begin a new editing session. If you are editing or adding to one file and you wish to clear what you have done and start over quickly, BEGIN will save time and add convenience. If the file you are editing was initiated with the FILE command, RISE will automatically close that file for you and save your changes so you can start working on a new file.

If you are editing a work file initiated by the TEXT command, the BEGIN command will clear the work file by purging it so that you may start over on a new file. Because you may wish to save the changes in a permanent file first, a safety prompt will remind you that you have not used the KEEP command with this file, and that it will be cleared upon execution. Thus you may reconsider the consequences of the command before consenting to them. With the NOW parameter in effect, however, RISE will clear the work file immediately, without displaying the safety messages.

Execution mode: Line

Examples

The following are legal abbreviations which may be used with the BEGIN command and its parameter:

BEG	Begin a new editing session.
BN	Disable safety prompts and immediately terminate the current editing session to begin a new one.

In the first example, the ADD command is used to create a five-line file. When the BEGIN command is issued, safety prompts are returned by RISE. An affirmative response causes the command to be executed (1). The results of BEGIN are shown when LIST ALL is commanded. The execution message informs the user that there is no edit file since the file has been cleared (2). Again the ADD command is used to create a file. In this case, the new file is made permanent using the KEEP command (3). Note that the new filename, "SAVEFILE", already exists, and a safety prompt appears to ensure that the user wishes to replace the old SAVEFILE with the new (4).

```
    >ADD
     Creating work space file.
        1    :LINE 1
        2    :LINE 2
        3    :LINE 3
        4    :LINE 4
        5    :LINE 5
        6    :<Control Y>
    >BEGIN
     KEEP not done, CLEAR current work file?YES
(1)  Ready to begin new session.
    >LIST ALL

(2)  Error 65: There is no edit file.
    >ADD
     Creating work space file.
        1  :LINE ONE
        2  :LINE TWO
        3  :LINE THREE
        4  :LINE FOUR
        5  :LINE FIVE
        6  :<Control Y>

(3) >KEEP SAVEFILE
     File exist already, purge old SAVEFILE?YES
     Keep completed.

(4)  File name is SAVEFILE
```

In the next example, SIMCAL is copied into a work file, and the first five lines are listed (1). When BEGIN is entered, no safety prompts appear because no changes have been made to the work file. With the ADD command, RISE is able to start a new file. Three lines are entered before <Control Y> ends Add Mode (2). The new file is then permanently stored under the filename "DATAFILE" (3).

```
    >TEXT SIMCAL
     Text completed.
    >LIST 1/5
        1     00011 H
        SIMCAL
        2     00012FINPUT         IP   F       72                DISK
        3     00013FOUTPUT        O    F       72                DISK
        4     00014IINPUT         AA   01    1 CA
(1)     5     00015I             OR    02    1 CS
    >
    >
    >BEGIN
     Ready to begin new session.
    >ADD
     Creating work space file.
        1    :DATA RECORD 1
        2    :DATA RECORD 2
        3    :DATA RECORD 3

(2)     4    :<Control Y>
    >KEEP DATAFILE

(3)  Keep completed.
     File name is DATAFILE
    >
```

In the next example, the FILE command is used to allow for direct editing of SIMCAL (1). Since SIMCAL is not a KSAM file and must be converted to one (see FILE), an execution message is returned which prompts the user for additional information (2). The file DIRSIM is opened, and 6 lines are listed (3). A CHANGE command is issued which changes the string "DISK" to "TAPE", and the results are listed (4).

Suppose at this point the user wishes to create a new file to contain important data. Currently, however, the user is directly editing DIRSIM. Before creating the new file, the user would want to close DIRSIM, making the new changes permanent and start a new editing session to create the new data file. The BEGIN command will accomplish this. When it is executed, an execution message is displayed (5), and a LIST ALL command brings an error message to the screen which indicates that RISE is ready for a new editing session (6).

```
    >
(1) >FILE SIMCAL

(2)  Enter new file name followed by additional number of records.
    DIRSIM 10
     File opened.
    >LIST 1/6
        1     00011H

(3)     SIMCAL
        2     00012FINPUT   IP   F       72           DISK
        3     00013FOUTPUT  O    F       72           DISK
        4     00014IINPUT   AA   01    1 CA
        5     00015I       OR    02    1 CS
        6     00016I       OR    03    1 CM

(4) >CHANGE ;DISK; TO ;TAPE; IN ALL
        2     00012FINPUT   IP   F       72           TAPE
        3     00013FOUTPUT  O    F       72           TAPE
     Changes  completed.

(5) >BEGIN
     Ready to begin new session.
    >LIST ALL

(6)  Error 65: There is no edit file.
```

The user is now able to create the new data file mentioned in the last example. The ADD command is given, and data records are added (1). The work file is stored to a permanent file with the KEEP command. The filename, DATAFILE, already exists, so RISE issues a safety prompt (2).

Finally, DIRSIM is again accessed for direct editing with the FILE command (3). The first 6 lines are listed (4). The listing shows that the changes were made permanent to DIRSIM when "DISK" was changed to "TAPE" (4) previously.

```
    >ADD
(1)  Creating work space file.
        1    :DATA RECORD ONE
        2    :DATA RECORD TWO
        3    :DATA RECORD THREE
        4    :<Control Y>
    >K DATAFILE

(2)  File exist already, purge old DATAFILE?YES
     Keep completed.
     File name is DATAFILE
    >
    >

(3) >FILE DIRSIM
     File opened.

(4) >L1/6
        1     00011H
        SIMCAL
        2     00012FINPUT   IP   F      72              TAPE
        3     00013FOUTPUT  O    F      72              TAPE
        4     00014IINPUT   AA   01   1 CA
        5     00015I       OR    02   1 CS
        6     00016I       OR    03   1 CM
    >
    >
    >
```

CHANGE

CHANGE changes an "oldstring" to a "newstring".

Form

```
CH[ANGE] "oldstring" TO "newstring" [IN lb [/le]]
```

Parameter Explanation

"oldstring" The old string you are replacing.

"newstring" The new string that is replacing the old.

IN Indicates that a range of lines to be changed follows.

lb First, or only, line number where change will be made.

le Line where the change ends.

Purpose

The CHANGE command replaces "oldstring" with "newstring". If no line range is specified, the changes will be made from the current position of the record pointer to the end of file.

Special Considerations

The number of characters in "oldstring" must be the s ame as that of "newstring". If there is any difference, the column position of characters would be changed, possibly making the RPG source line syntactically incorrect.

Delimiters

You may use any special characters as delimiters except the slash (/).

Execution mode: Line or Block.

Record pointer: At last line changed.

Examples

The following are legal abbreviations which may be used with the CHANGE command and its parameters:

```
CH;OLD;TO $NEW$  Change oldstring TO newstring from first line to current
IN F/*           position of record pointer.

CHAN;OLD; TO;    Change oldstring to newstring in lines 1 through 10.
NEW;IN1/10
```

In the examples of the CHANGE command, a mistake is made in the spelling of the command, and an error message is returned (1).

Next, there is an attempt to replace a three character string, ";PUT;" with a four character string ";FILE;", and another error message is generated (2).

When an additional space is added between the delimiters containing the oldstring ";PUT ;" so that both strings are equal in length, the CHANGE is executed (3).

```
    >LIST ALL
        1          H
        SIMCAL
        2          FINPUT   IP  F      72            DISK
        3          FOUTPUT  O   F      72            DISK
        4
        5          IINPUT   AA  01   1 CA
        6          I       OR   02   1 CS
        7          I       OR   03   1 CM
        8          I       OR   04   1 CD
        9          I       OR   05
       10          I                             8   72OPRND1
       11          I                             9  132OPRND2         99

       12
(1) >CHANGE ;PUT; TO ;FILE; IN ALL
          ^
     Error 9: Syntax error on CHANGE command.

(2) >CHANGG ;PUT; TO ;FILE; IN ALL
     Error 109: Oldstring length must equal newstring length.

(3) >CHANGE ;PUT ; TO ;FILE; IN ALL
        2          FINFILE      IP  F      72            DISK
        3          FOUTFILE     O   F      72            DISK
        4          IINFILE      AA  01   1 CA
```

COMMENT

COMMENT produces a comment banner to contain the comments you enter.

Form

```
     COM[MENT] [C] [lb] [BY inc]
```

Parameter Explanation

C — A fifteen line comment banner is displayed. (If the C parameter is not given, the default banner occupies 10 lines.) The "C" parameter displays a banner with the Column Indicator form instead of the RPG Record Specification form. A space is not required between this parameter and the command.

lb — Location where comment banner is to be added. If this parameter is omitted, the banner is added at the end of the work file. If "lb" exists, the destination will be changed to the next sequential line.

BY inc — Increment value for each line in the comment banner.

Purpose

The COMMENT command allows you to enter comments in your s ource program wherever you wish. When the command is entered, RISE shifts you to SHOW mode and automatically provides a banner, a rectangle composed of asterisks. With it is the RPG Record Specification form corresponding to the line preceding the banner. If "C" is specified, a Column Indicator is shown with the line instead.

Special Considerations

When you wish to delete your COMMENT, you may use the DELETE command and specify the sequence range. You may also display the banner in SHOW mode and press the terminal's Clear Display key followed by the ENTER key.

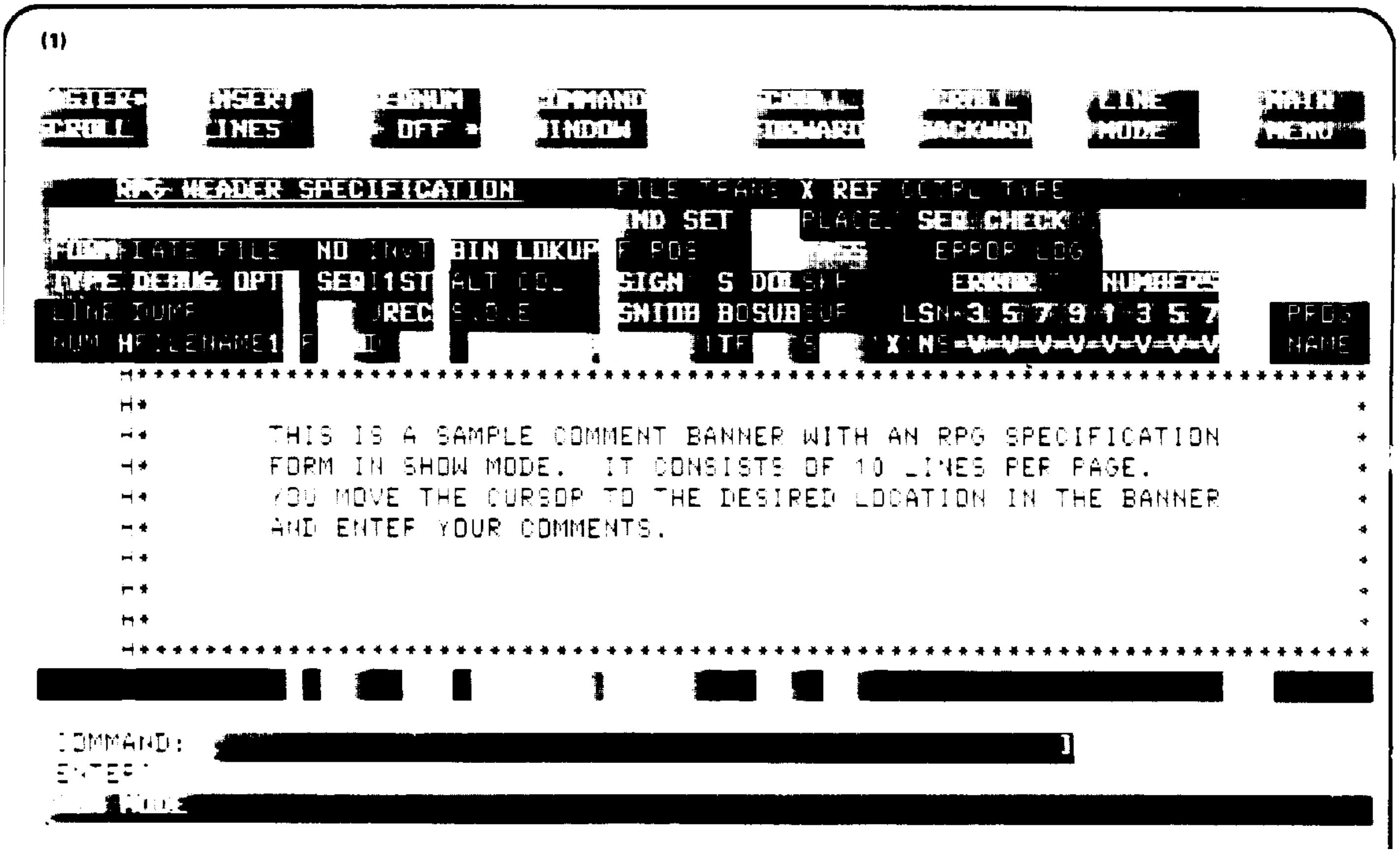
Execution mode: Block

Examples

The following are legal abbreviations which can be used with the COMMENT command and its parameters:

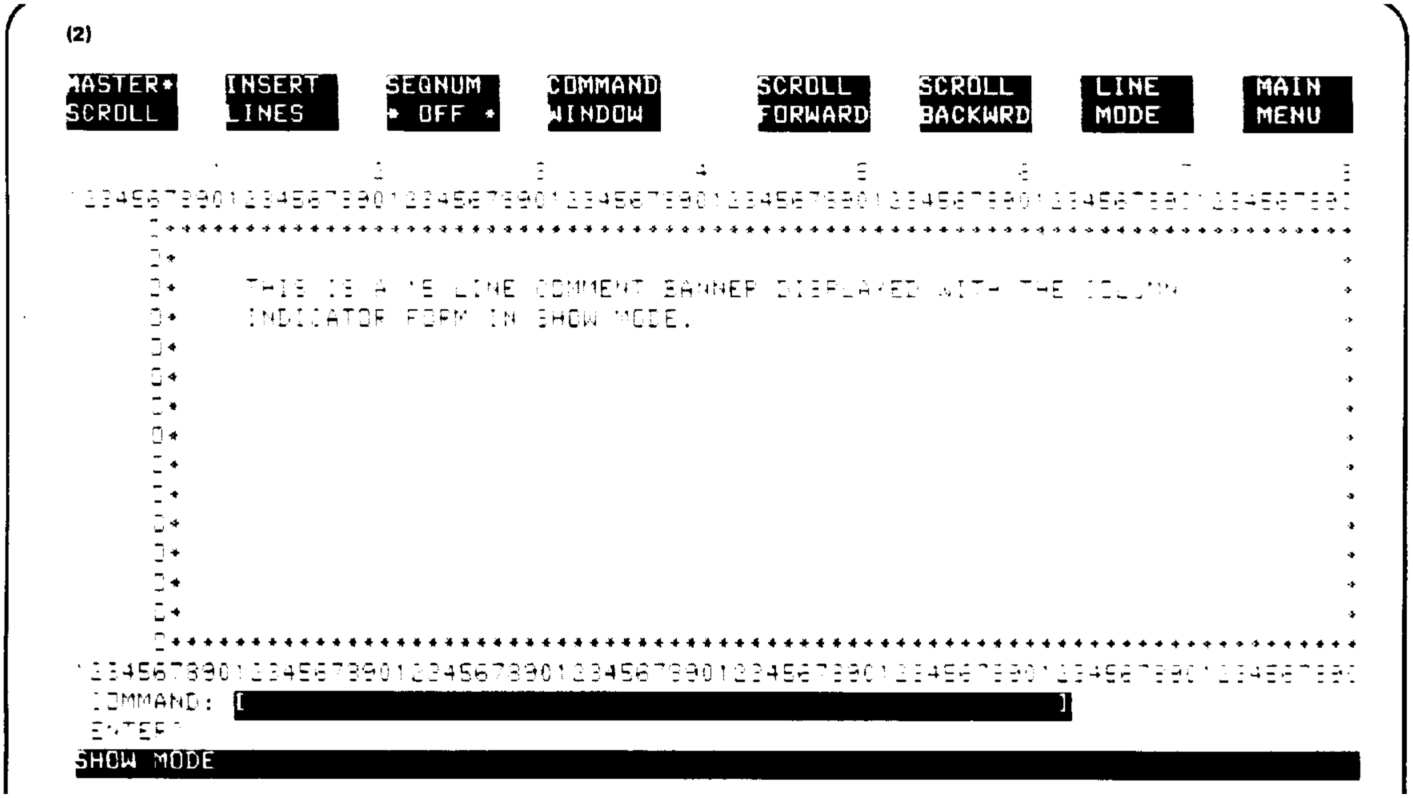
```
COMC FIRST BY 1  Display fifteen line banner beginning at first line with
                 increment value of 1.

COM 20           Display ten line comment banner beginning at line 20.
```

The first example shows a ten line comment banner in Show Mode. It is displayed with the Header Specification form (1).



```
H***********************************************************************
H*                                                                     *
H*      THIS IS A SAMPLE COMMENT BANNER WITH AN RPG SPECIFICATION      *
H*      FORM IN SHOW MODE.  IT CONSISTS OF 10 LINES PER PAGE.          *
H*      YOU MOVE THE CURSOR TO THE DESIRED LOCATION IN THE BANNER      *
H*      AND ENTER YOUR COMMENTS.                                       *
H*                                                                     *
H*                                                                     *
H*                                                                     *
H***********************************************************************
```

The next example is of the fifteen line comment banner shown with the column indicator (2).



```
         1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890
     C***********************************************************************
     C*                                                                     *
     C*    THIS IS A 15 LINE COMMENT BANNER DISPLAYED WITH THE COLUMN       *
     C*    INDICATOR FORM IN SHOW MODE.                                     *
     C*                                                                     *
     C*                                                                     *
     C*                                                                     *
     C*                                                                     *
     C*                                                                     *
     C*                                                                     *
     C*                                                                     *
     C*                                                                     *
     C*                                                                     *
     C*                                                                     *
     C***********************************************************************
12345678901234567890123456789012345678901234567890123456789012345678901234567890
```

COPY

COPY allows you to duplicate lines in a file.

Form

```
     COP[Y] lb [/le] TO loc [BY inc]
```

Parameter Explanation

lb First--or only--line you wish to have copied.

le Last line of a group of lines you are copying elsewhere.

TO loc Location where the lines will be copied. The location cannot be within the range of lines specified "lb/le" after COPY. If the destination line exists, the lines will be copied starting at the next line in sequence.

BY inc Increment value by which you want the lines copied. If the number of lines to be copied cannot fit from line "loc" to the next existing higher line number, the copy is not performed. Using an increment value of less than 1 allows you to concentrate more lines in a restricted location. RISE will automatically calculate a smaller increment value to concentrate more lines in a restricted area when it is possible.

Purpose

The COPY command allows you to duplicate lines from one location to a new location in the work file. The original lines remain where they were after the COPY command is executed.

```
 *  Execution mode:  Line or Block.

 *  Record Pointer:  At last original line that was duplicated.

 *  Control Y:    Stops the listing of lines that are being copied.
```

Examples

The following are legal abbreviations which may be used with the COPY command and its parameters:

```
COP 10 TO LAST   Duplicate line 10 beginning after the last line of the existing file.

COP1/10          Copy lines one through ten to line twenty with an
TO20BY.2         increment value of 0.2.
```

In the first example, an attempt to copy lines to a location within the range of lines being copied generates an error (1).A second error results when the range of lines to be copied is incorrectly entered (2). The third error occurs when no "TO" location is specified for the copy command (3).

Finally, a legal command causes lines 1/5 to be duplicated starting at the first line following the last line in the existing file (4).

```
     (1) >COPY  1/10 TO 5
          Error 151:  TO location cannot be within line range.

     (2) >COPY 5/1 TO 10
```

```
          Error 62: The TO line cannot be less than the FROM line number.

     (3) >COPY ALL
          Error 149: TO location missing.

     (4) >COPY 1/5 TO LAST BY 1
            39     00011H
             SIMCAL
            40     00012FINPUT   IP   F      72            DISK
            41     00013FOUTPUT  O    F      72            DISK
            42     00014IINPUT   AA   01  1  CA
            43     00015I       OR    02  1  CS
          Copy completed.
         >
```

DELETE

DELETE enables you to delete lines from a work file.

Form

```
     D[ELETE] lb[/le]
```

Parameter Explanation

lb The first or only line number where you wish to start your deletion.

le The last line you are deleting.

Purpose

The DELETE command is used to delete a line or range of lines from a work file. Note: when in Block Mode, you should have the sequence numbers on the screen so that you can see the sequence numbers of the lines you are deleting. Use special function key F-3 to display the numbers. Do not use the RPG line numbers to establish a line range for deletions.

Related Commands

The UNDEL command restores lines that were mistakenly deleted. Under MODIFY, the Delete subcommand interactively deletes characters within a line.

Execution mode: Line or Block.

Record pointer: At the line following the last deleted line.

Control Y: Stops the listing of deleted lines. Does not stop the deletion.

Examples

The following are legal abbreviations which may be used with the DELETE command and its parameters:

D10/50 Delete lines 10 through 50.

D10 Delete line 10.

```
DELE A           Delete all lines in the file.
```

In the example, an error is produced when an attempt to delete nonexistent lines is made (1).A second error occurs when ALL is misspelled (2). A third results from an unnecessary character entered following last line (3).Finally, those lines copied to the end of SIMCAL in the COPY example above are deleted by a legal command, and the results are displayed (4).

```
          >
          >
     (1)  >DELETE 200/300
           Error 81: Range is empty.

     (2)  >D ALLL
                ^
           Error 67: Syntax error on ALL option.

     (3)  >D F/LT
                ^
           Error 71: Syntax error on Last.

     (4) >D39/LAST
            39      00011H
             SIMCAL
            40      00012FINPUT   IP  F     72           DISK
            41      00013FOUTPUT  O   F     72           DISK
            42      00014IINPUT   AA  01  1  CA
            43      00015I       OR   02  1  CS
          Delete completed.
         >
```

EXIT

EXIT terminates execution of RISE.

Form

```
     E[XIT] [N[OW]]
```

Parameter Explanation

N[OW] Exit immediately without keeping the work file and with safety prompts disabled. If you are directly editing your file, using the FILE command, the EXIT command will close the file, keeping your changes. If the file being edited is a copy of the original, initiated by the TEXT or ADD commands, the EXIT command will not terminate RISE unless a KEEP command is given to protect your recent changes. The NOW parameter allows you to EXIT immediately and discards any changes you may have made to the file. If you choose not to use the "NOW" parameter, RISE will return a message asking if it is all right to clear the work file upon termination. If you respond with "Y[ES]", it will clear the file and exit. If you respond with "N[O]", it will cancel the command so that you can perform a KEEP first if you wish.

Purpose

The EXIT command terminates the execution of RISE, returning control to MPE.

Execution mode: Line or Block.

Examples

The following are legal abbreviations which may be used with the EXIT command and its parameters:

```
EXI NOW          Exit immediately without the safety prompts.

E N              Exit immediately without the safety prompts.

E                Exit--safety prompts may be displayed on the screen.
```

In the example, a misspelling of EXIT causes an error message to be displayed (1). EXIT is then entered correctly, and a safety prompt reminds the user that the KEEP command has not been entered and asks if clearing the work file is acceptable. The user responds negatively to the prompt, and the command is cancelled (2).

An unacceptable parameter--MAYBE--produces another error (3). An illegal special character is introduced between EXIT and the NOW parameter, with another error resulting (4).Finally, EXIT NOW is entered, and the command is completed with the execution message displayed (5). The colon is MPE's prompt.

```
         >
     (1) >EXT
            ^
          Error 15: Syntax error on EXIT command.

     (2) >EXIT
          KEEP not done, CLEAR current work file?NO
          Exit cancelled.

     (3) >EXIT MAYBE
               ^
          Error 34: Only NOW parameter allowed with Exit command.

     (4) >EXIT,NOW
              ^
          Error 36: Ilegal special character

     (5) >EXIT NOW

          END OF PROGRAM
          :
```

FILE

FILE allows you to edit a file directly.

Form

```
     FIL[E] filename [N[OW]]
```

Parameter Explanation

filename — Name of the file to be edited directly.

N[OW] — Execute the command immediately without safety prompts. NOW is significant with FILE if, previous to issuing the command, you were editing a copy of another file. If you were directly editing another file, having used the FILE command, that file will be closed automatically, and your changes made permanent. However, if you used the TEXT command to copy it into a work file, RISE will not open another file unless you enter the KEEP command to save your changes, or specify the NOW parameter to discard your changes. The NOW parameter is used when immediate execution is required. It disables safety prompts.

If you choose not to use the "NOW" parameter, RISE will return a safety prompt asking if you want to clear the current work file. If you respond with "Y[ES]", it will clear the file, and another file will be accessed for editing. If you respond with "N[O]", it will cancel the FILE command so that you may perform a KEEP first if you wish.

Purpose

The FILE command allows you to edit directly a file named "filename". If "filename" is not already a KSAM file, RISE will create one for editing and prompt you for maximum number of records in the new file. In other words, the KSAM work file is your permanent source file.

FILE is especially useful when your source program is long. If you use the TEXT/KEEP commands to bring the file to the terminal for editing or additions, this means that a permanent file must be copied into a work file (KSAM file). Such copying increases overhead. The FILE command does not require a file to be copied.

A disadvantage of the FILE command is that an error that occurs during editing cannot be corrected easily and quickly unless a backup copy was made for purposes of restoration before the editing began.

The FILE command specifies that the file "filename" is the file which is to be edited directly. RISE then ensures that "filename" is in the KSAM file form. To do so, three possible cases must be considered.

Case 1. Creating New File

The file "filename" does not exist and must be created as a KSAM file. In order to do this, you create the file.

```
You type        > FILE "filename " **
```

RISE responds Creating new KSAM file, enter maximum number of records.

Case 2. Accessing Old KSAM File

"filename" has been created as a KSAM file (called "newfile" below).

```
You type        > FILE "newfile "
```

RISE responds File opened.

** You do not enclose the filename in quotes when entering a command. Quotes are used around filename in these cases to indicate a place holder representation of the real name of your file. For example, FILE "filename" could represent FILE SIMCAL.

Case 3. Converting Non-KSAM File to New KSAM File

"filename" exists but is not a KSAM file. RISE must convert it. (The converted file is called "newfile" below.)

```
You type         > FILE "filename "
```

RISE responds Enter new file name followed by additional number of records.

```
You type         "newfile " 100
```

(If you are adding records to those in "filename" (now called "newfile") enter the number of additional records you will add--100 in this example. The default is the number of records in "filename".)

To access converted files in the future, type FILE "*newfile* ".

Related Commands

The TEXT and KEEP commands achieve results similar to those of FILE and EXIT TEXT and KEEP have the disadvantage of additional overhead while files are being copied to a work file and recopied to a permanent file. The advantage is that any erroneous editing of the work file pertains only to the work file. The permanent file remains in its previous state. See TEXT and KEEP for further information concerning these commands.

It is recommended that TEXT and KEEP be used with small files since their copy time will be minimal. FILE and EXIT are best suited to large files in their developmental stage. When the source program is fully developed and in production, use TEXT and KEEP to avoid key file overhead in storing.

A KSAM file actually consists of two files: the data file and the key file. If you want to purge both parts of a KSAM file, use the following commands:

```
       :RUN KSAMUTIL.PUB.SYS
       >PURGE "filename"
       >EXIT
```

For further information concerning KSAM files, please see the *KSAM REFERENCE MANUAL*.

Leaving FILE

Use the EXIT command to leave FILE. It will terminate RISE and return you to MPE with your changes made permanent. You can also use the BEGIN command to leave FILE. It will close the file you are directly editing and allow you to begin a new editing session as if you had just entered the RPG Interactive System Environment.

```
 *  Execution mode:  Line or Block.

 *  Record Pointer:  At first line in file.
```

Examples

The following are legal abbreviations which may be used with the FILE command and its parameters:

```
FILE FILE1 N     Ready a work file named FILE1 for immediate editing.

FIL FILE1        Ready a work file named FILE1 for editing.
```

In the first example, the FILE command is used to create a new KSAM file named SIMCAL2. A message is returned asking for the maximum number of records to be included in the new file. The number given is 5 (1). This is an example of Case 1 (see above).

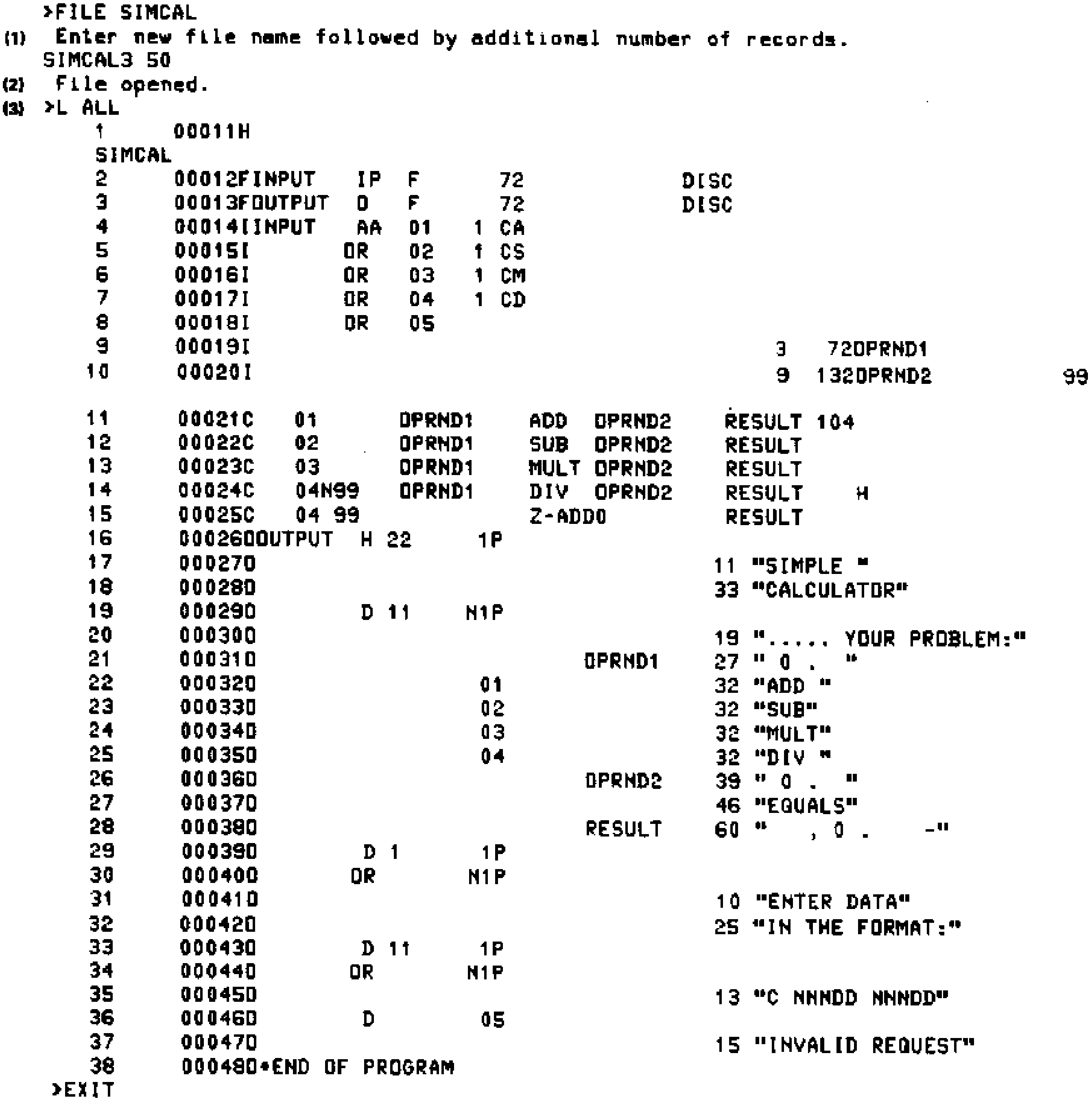
The ADD command is entered, and an attempt to add the 6th record is nullified. The error message gives the reason and recommends using the XPAND command to increase file size (2). When XPAND is entered without the number of records to expand the file by, another error message appears (3).

The XPAND command is again tried, this time with the maximum number of records specified alphabetically rather than numerically. This generates another error message (4).

When the XPAND command is properly entered, an execution message is displayed (5). The ADD that follows is accepted by RISE, and lines are added before <Control Y> is applied to exit Add mode (6).

```
    >FILE SIMCAL2
(1)  Creating new Ksam file, enter maximum number of records.
    5
     File opened.
    >ADD
        1    :LINE 1
        2    :LINE 2
        3    :LINE 3
        4    :LINE 4
        5    :LINE 5
        6    :LINE 6

(2)  Error 147: No room in edit file, use XPAND.

(3) >XPAND
     Error 137: Missing number of records to expand by.

(4)  >XPAND FIFTY
            ^
     Error 138: Illegal numeric value.

(5) >XPAND 50
     Xpand completed.
    >ADD
        5    :LINE 5
        6    :LINE 6
        7    :LINE 7
        8    :<Control Y>

(6) >EXIT
```

In the second example, which typifies Case 3 (above), SIMCAL exists but is not a KSAM file, and RISE must convert it. A message in response to the FILE command asks for a new file name and a maximum number of records (1). When the filename and record number are given, the file is opened (2). The converted file is then listed (3).



```
   >FILE SIMCAL
(1) Enter new file name followed by additional number of records.
   SIMCAL3 50
(2) File opened.
(3) >L ALL
       1     00011H
       SIMCAL
       2     00012FINPUT   IP  F      72            DISC
       3     00013FOUTPUT  O   F      72            DISC
       4     00014IINPUT   AA  01   1 CA
       5     00015I       OR   02   1 CS
       6     00016I       OR   03   1 CM
       7     00017I       OR   04   1 CD
       8     00018I       OR   05
       9     00019I                                         3   72OPRND1
      10     00020I                                         9  132OPRND2          99

      11     00021C    01     OPRND1     ADD  OPRND2   RESULT 104
      12     00022C    02     OPRND1     SUB  OPRND2   RESULT
      13     00023C    03     OPRND1     MULT OPRND2   RESULT
      14     00024C    04N99  OPRND1     DIV  OPRND2   RESULT    H
      15     00025C    04 99             Z-ADD0        RESULT
      16     000260OUTPUT  H 22    1P
      17     000270                                    11 "SIMPLE "
      18     000280                                    33 "CALCULATOR"
      19     000290         D 11    N1P
      20     000300                                    19 "..... YOUR PROBLEM:"
      21     000310                           OPRND1   27 " 0 .  "
      22     000320                 01                 32 "ADD "
      23     000330                 02                 32 "SUB"
      24     000340                 03                 32 "MULT"
      25     000350                 04                 32 "DIV "
      26     000360                           OPRND2   39 " 0 .  "
      27     000370                                    46 "EQUALS"
      28     000380                           RESULT   60 "    , 0 .    -"
      29     000390         D 1      1P
      30     000400        OR      N1P
      31     000410                                    10 "ENTER DATA"
      32     000420                                    25 "IN THE FORMAT:"
      33     000430         D 11     1P
      34     000440        OR      N1P
      35     000450                                    13 "C NNNDD NNNDD"
      36     000460         D        05
      37     000470                                    15 "INVALID REQUEST"
      38     000480*END OF PROGRAM
    >EXIT
```

The third example of the FILE command usage represents Case 2, in which an old KSAM file (SIMCAL2, which was created in the first example) is made available for direct editing using the FILE command. RISE responds with a message stating that the file is opened (1). LIST ALL shows the contents of the file (2). Exit is used to leave RISE (3).

```
        >FILE SIMCAL2
    (1)  File opened.

    (2) >L A
             1     LINE 1
        2     LINE 2
        3     LINE 3
        4     LINE 4
        5     LINE 5
        6     LINE 6
        7     LINE 7

(3) >EXIT

    END OF PROGRAM
    :
```

In the fourth example, the TEXT command is used with SIMCAL, and the first five lines are listed (1). Next, the five lines are deleted (2). When the user attempts to change work files to SIMCAL3, using the FILE command, a safety prompt is displayed. The negative response results in a cancellation of the FILE command (3).

SIMCAL, with the first five lines deleted, is then copied into a permanent file named SAVEFILE using the KEEP command (4). Two execution messages are displayed (5). A DELETE ALL command is given. Note that line 6 is the first in the file (6). <Control Y> is applied to halt the listing of the deleted lines but, as the message indicates, it does not halt the deletion (7). The FILE SIMCAL3 command, this time with the NOW parameter added to disable the safety prompts, is executed (8).

```
    >TEXT SIMCAL
     Text completed.
    >
(1) >L 1/5
        1     00011H
        SIMCAL
        2     00012FINPUT   IP  F        72                DISK
        3     00013FOUTPUT  O   F        72                DISK
        4     00014IINPUT   AA  01    1  CA
        5     00015I       OR   02    1  CS

(2) >D 1/5
        1     00011H
        SIMCAL
        2     00012FINPUT   IP  F        72                DISK
        3     00013FOUTPUT  O   F        72                DISK
        4     00014IINPUT   AA  01    1  CA
        5     00015I       OR   02    1  CS
     Delete completed.
    >
    >FILE SIMCAL3

(3)  KEEP not done, CLEAR current work file?NO
     File cancelled.

(4) >KEEP SAVEFILE
```

```
(5)  Keep completed.
     File name is SAVEFILE

(6) >D ALL
        6     00016I        OR  03    1  CM
        7     00017I        OR  04    1  CD
        8     00018I        OR  05
        9     00019I                     8   72OPRND1
       10     00020I                     9  132OPRND2            99

       11     00021C   01      OPRND1       ADD  OPRND2    RESULT 104
       12     00022C   02      OPRND1       SUB  OPRND2    RESULT
       13     00023C   03      OPRND1       MULT OPRND2    RESULT
       14     00024C   04N99   OPRND1       DIV  OPRND2    RESULT     H
       15     00025C   04 99             <Control Y>

(7)  Delete completed.

(8) >FILE SIMCAL3 NOW
     Fi1e opened.
    >EXIT

    END OF PROGRAM
```

The last example shows how to purge files created by the FILE command as well as the key files associated with KSAM files. First KSAMUTIL.PUB.SYS is run, and an HP banner is displayed (1). The user then enters the file names of the files with the PURGE command (2). Note that the execution messages contain those filenames plus the RISE-created file designators of the associated key files in the execution messages (3).

The EXIT ends the KSAM UTIL program (4).

```
(1)  :RUN KSAMUTIL.PUB.SYS

     HP32208A.03.03  WED,  MAY 27,  1981,  3:55 PM  KSAMUTIL  VERSION:A.03.03

     HP32208A.03.03  WED,  MAY 27,  1981,  3:55 PM  KSAMUTIL  VERSION:A.03.03

(2)  >PURGE SIMCAL2

(3)  SIMCAL2.RPG.SUBSYS & SIMCA97 PURGED.
     >PURGE SIMCAL3
     SIMCAL3.RPG.SUBSYS & SIMCA18 PURGED.

(4)  >EXIT
     END OF PROGRAM
     :
```

FIND

FIND locates the first or every occurrence of a string in a work file.

Form

```
     F[IND] [E[VERY] ] "string" [IN lb[/le] ]
```

Parameter Explanation

E[VERY] Every occurrence of the string is located in the given range. (If not specified, only the first occurrence is located.)

"string" Character or characters you wish to find. (Note that you may omit "string", and RISE will default and use the previous "string" given with the last FIND command. However, you must specify "string" once.)

IN Notifies RISE that the next number or character will establish the range of the search for "string".

lb First or only line designated to be searched.

le Line where the search ends.

Purpose

The purpose of the FIND command is to locate the first occurrence of a string. If "EVERY" is specified, every occurrence of "string" will be located in the given range. If no range is given, RISE will search starting from the line after the current record pointer through to the end of the file. If "EVERY" is omitted, the command will list the first occurrence within the range. When you press "F" again, RISE will search for the next occurrence within the range. Stepping through each located line in this manner allows you to review it for any necessary modifications. When there is no occurrence of the string, RISE will print:

```
     "string" not found
```

Delimiters: The delimiters around "string" may be any special character except the slash (/).

Execution mode: Line or Block.

Record pointer: At the line containing "string".

Examples

The following are legal abbreviations which may be used with the FIND command and its parameters:

```
FIN "STRING" IN  Search the entire file for the first occurrence of
A                "STRING".  Each subsequent occurrence may be found each
                 time "F" is pressed until all are found.

F "STRING" IN    Find the first occurrences of "STRING" in a line range
10/50            beginning with line 10 and ending with line 50.

F EVERY          Search the entire file for all occurrences of "STRING"
"STRING" IN A    and list them at the terminal.
```

In the example, the TEXT command is used to copy SIMCAL into a work file, and lines 1 through 5 are listed (1). The FIND command is used with the EVERY parameter to find the string "PUT", and all occurrences of the string are listed (2). Next, FIND is again used with EVERY but this time the string is not specified. The default string, "PUT", which was used with the last FIND command, is used again by RISE, and the results are listed (3). An error is generated when an abbreviation of FIND cannot be distinguished from other commands which are similar (4). Note that "F" alone is defaulted to FIND, but when "F" is used with a second character, RISE requires a more thorough identification of the command, and no default is assumed.

Finally, FIND is used with the string "DISK" in a line range. The first occurrence of "DISK" is listed on the screen (5). When the "F" command is given, the next occurrence is listed (6). An execution message informs the user that all occurrences have been found (7).

```
     >TEXT SIMCAL
(1)   Text completed.
     >LIST 1/5
         1     00011H
         SIMCAL
         2     00012FINPUT   IP  F       72               DISK
         3     00013FOUTPUT  O   F       72               DISK
         4     00014IINPUT   AA  01   1  CA
         5     00015I       OR   02   1  CS
     >

(2)  >FIND EVERY ;PUT; IN ALL
         2     00012FINPUT   IP  F       72               DISK
         3     00013FOUTPUT  O   F       72               DISK
         4     00014IIN‘PUT"   AA  01   1   CA
        16     00026OOUTPUT  H 22     1P
     >

(3)  >F EV IN A
         2     00012FINPUT        IP  F       72           DISK
         3     00013FOUTPUT       O   F       72           DISK
         4     00014IINPUT        AA  01   1  CA
        16     00026OOUTPUT  H 22     1P
     >
     >

(4)  >FI ;DISK; IN 1/10
        ^
      Error 19: Can't distinguish between FILe, FINd, or FOrm.
     >

(5)  >F ;DISK; IN 1/10
         2     00012FINPUT   IP  F       72              DISK

(6)  >F
         3     00013FOUTPUT  O   F       72              DISK

(7)  >F
      'DISK' not found.
     >
```

>

FORM

FORM displays an RPG Record Specification form or column indicator on the screen.

Form

```
    FO[RM] form
```

Parameter Explanation

"form"	Meaning
H	Header specifications
F	File specifications
E	File extension specifications
L	Line counter specifications
I	Input specifications
C	Calculation specifications
O	Output specifications
CO[L]	Column indicator

NOTE Space is not required between command and parameters.

Purpose

The FORM command enables you to display an RPG Record Specification form of the type "form" in Line mode. This gives you a visual "heading" for the form with abbreviated explanations of the entries expected for the columns. A column indicator is also available with the "COL" option.

Related Commands

While in ADD mode you can change forms by entering:

```
    ^^ form
```

Execution mode: Line

Examples

The following are legal abbreviations which may be used with the FORM command and its parameters:

FORMH	Display Header Specification form.
FOO	Display Output Specification form.
FOCOL	Display column indicator.

In the first example, a non-existent form is called for, using the FORM command. It generates an error message (1). The command is then entered correctly for the Column Indicator to be displayed (2).

The Column Indicator is displayed and SIMCAL, which is the work file, is listed (3). <Control Y> halts the listing (4).

```
          >
          >FORM XSPECS
     (1)        ^
           Error 173: Only H,F,E,L,I,C,O, or COl allowed.

     (2)  >FORM COL

                   0         1         2         3         4         5        6
        7

12345678901234567890123456789012345678901234567890123456789012345678901234567890

     (3)  >L A
              1     00011H
              SIMCAL
              2     00012FINPUT   IP  F       72           DISK
              3     00013FOUTPUT   O  F       72           DISK
              4     00014IINPUT   AA  01   1  CA
              5     000151        OR  02   1  CS
              6     00016I        OR  03   1  CM
              7     00017I        OR  04   1  CD
              8     00018I        OR  05
              9     00019I                                        8   72OPRND1
             10     00020I                                        9  132OPRND2
       99

     (4)  <Control Y>
          >
```

In the second example, an error is generated when FORM is entered without a particular form specified (1). The FORM command is then correctly entered, with the Header Specification form specified, and the form is displayed (2). The first line of SIMCAL is listed and shown in relationship to the Header Specification form (3).

```
          >
     (1)  >FORM
           Error 173: Only H,F,E,L,I,C,O, or COl allowed.

     (2)  >FORMH
```

(Graphic is unreadable #201G.tif).

GET

GET is used when you wish to execute a sequence of commands previously stored in a file.

Form

```
     G[ET] filename
```

Parameter Explanation

filename   The name of the file containing the commands.

Purpose

The GET command is used when you have a number of commands stored in a file which you wish to have executed again at another time. Using "GET filename", you first have the commands displayed at the terminal. They are then executed automatically until the End of File (EOF) of the command file is reached.

The command file--the file whose commands "GET" executes--is where you imbed special specifications for the command. For example, a comment may appear in the command file by specifying an asterisk (*) as the first character in a new line.

You may use the GET command to activate the softkeys to execute commands. The function of a particular softkey may be accomplished indirectly by putting the numbers 0 through 8 on a single line in the command file where 0=ENTER key, 1=F1 (special function key 1), 2=F2, up to 8=F8. Before entering an integer from 0 to 8, you must enter a Block mode command so that the softkeys will be activated.

You may wish to review the commands that are displayed on the screen before they are automatically executed. You do so by setting pauses to delay execution. To do this, you specify "#n" where "n" equals the number of seconds of pause you require. If "n" is not specified, the pause is turned off. It stays in effect on every command until you change it with another "#n". The maximum value of "#n" is approximately 2,147,484 seconds.

Additionally, "%" on a new line will suspend the operation of RISE until any of the special function keys is pressed while in Block Mode or until the Carriage Return key is pressed while in Line Mode. When the appropriate continuation key is pressed, RISE will continue executing the remaining commands from the command file.

If the command file contains an ADD command, RISE will enter ADD mode. Control is then returned to you so that you can enter new text as desired. When you exit ADD mode, RISE will resume control and continue to execute commands from the command file. You cannot imbed new text in the command file and expect RISE to enter it the way it executes other commands. In this respect, ADD is not automatically governed by GET as other RISE commands are.

Similarly, if the command file contains a Modify command, RISE will enter Modify mode and then return control to you. You then type in your Modify subcommands from your terminal. RISE will not read and execute subcommands stored in your command file with the Modify command.

You cannot imbed the GET command within a command file. It can only be issued at the terminal.

NOTE An example of the use of the GET command is RISETOUR. You may text in RISETOUR.PUB.SYS and list it offline to a line printer to see an example of a command file.

Examples

The following is a legal abbreviation which may be used with the GET command and its parameter:

```
G CMDFILE         The RISE commands imbedded in a file named CMDFILE will
                  be read and executed.
```

As an example of a file that might be used with the GET command, a brief file is created. It is somewhat self-explanatory. Note the asterisks preceding the comments in lines 1/7 (1). In line 8, TEXT SIMCAL is entered. Following that is a command to LIST ALL (2). The pause in line 17 is incorrectly entered and must be modified to be acceptable to RISE (3) (5).

The SHOW FIRST command is imbedded, followed by a series of 5's entered from lines 18 through 23. If the file were being run, through the use of the GET command, SHOW FIRST would bring the Header Specification form to the screen. The number 5 would operate as if special function key #5 were being pressed, and you would Scroll Forward to the next Record Specification form after the pause had been completed (4). In line #26, <Control Y> is used to exit ADD mode. MODIFY is used to insert the pound sign (#) before the pause (5). CMDFILE is then stored as a permanent file using the KEEP command (6).

```
          >ADD
```

```
      Creating work space file.
 (1)      1    :*  THIS IS A SAMPLE COMMAND FILE.
          2    :*  AN ASTERISK INDICATES A COMMENT LINE.
          3    :*  YOU STORE A SEQUENCE OF COMMANDS IN A
          4    :*  FILE TO HAVE THEM ALL EXECUTED
          5    :*  AUTOMATICALLY.
          6    :*
          7    :*  TEXT IN A FILE.

 (2)      8    :TEXT SIMCAL
          9    :LIST ALL
         10    :*
         11    :*  LET'S USE SHOW MODE TO AUTOMATICALLY
         12    :*  SCROLL THROUGH THE FILE WITH THE RPG
         13    :*  SPECIFICATION RECORD FORMS.
         14    :*
         15    :*  WE SHOULD TURN THE PAUSE ON SO THAT WE
         16    :*  HAVE TIME TO REVIEW THE SPECIFICATIONS.

 (3)     17    :20

 (4)     18    :SHOW FIRST
         19    :5
         20    :5
         21    :5
         22    :5
         23    :5
         24    :EXIT NOW
         25    :*  ALL DONE WITH COMMAND FILE.
         26    :<Control Y>
     >

 (5) >M17
         17     20
               :I#
         17     #20
               :

 (6) >KEEP CMDFILE
      Keep completed.
      File name is CMDFILE
     >
     >
```

HELP

HELP displays information concerning RISE commands.

Form

```
H[ELP] [command]
```

Parameter Explanation

command — RISE command which you wish to have described in detail. You may use the abbreviation of the command so long as there is enough of it to distinguish it from other commands beginning with the same letters.

Purpose

The HELP command displays information concerning RISE commands on your screen. It will also provide a detailed description of a single command which you specify. If you use HELP without specifying a single command, brief descriptions of all RISE commands are displayed at the terminal.

Related Commands

Special function key (softkey) labels are displayed on the top of the screen with the HELP command. They have the following meanings:

Key	Meaning
F1	(not used)
F2	(not used)
F3	RPG source lines with or without editor sequence line numbers will be displayed with the next SHOW command you enter.
F4	Move cursor to Command Window.
F5	(not used)
F6	If you came from SHOW mode, return to last page that was displayed.
F7	Enter Line Mode.
F8	Display main menu.

Execution mode: Block.

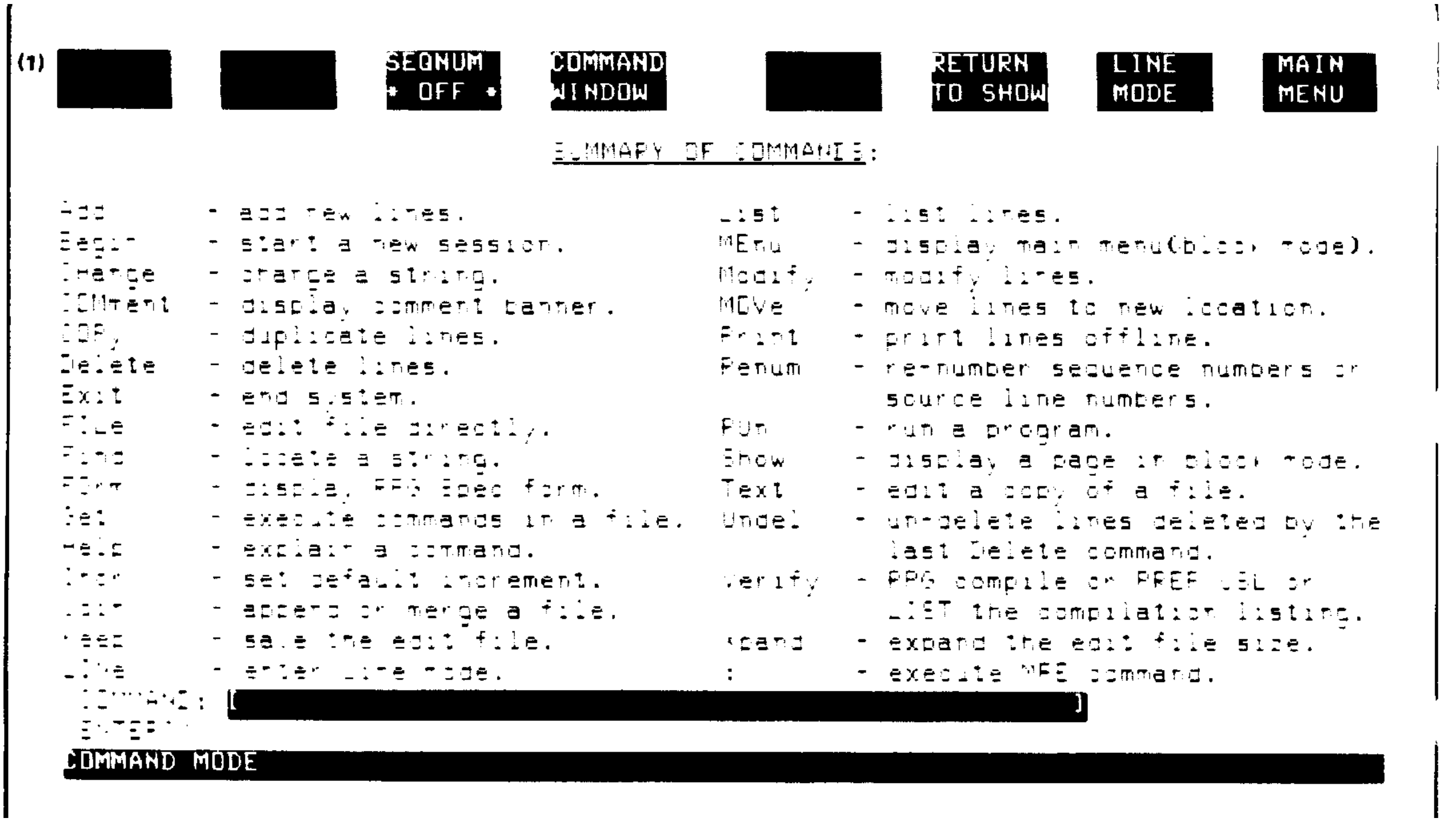
Examples

The following are legal abbreviations which may be used with the HELP command and its parameters:

HL An explanation of the LIST command will appear on the screen.

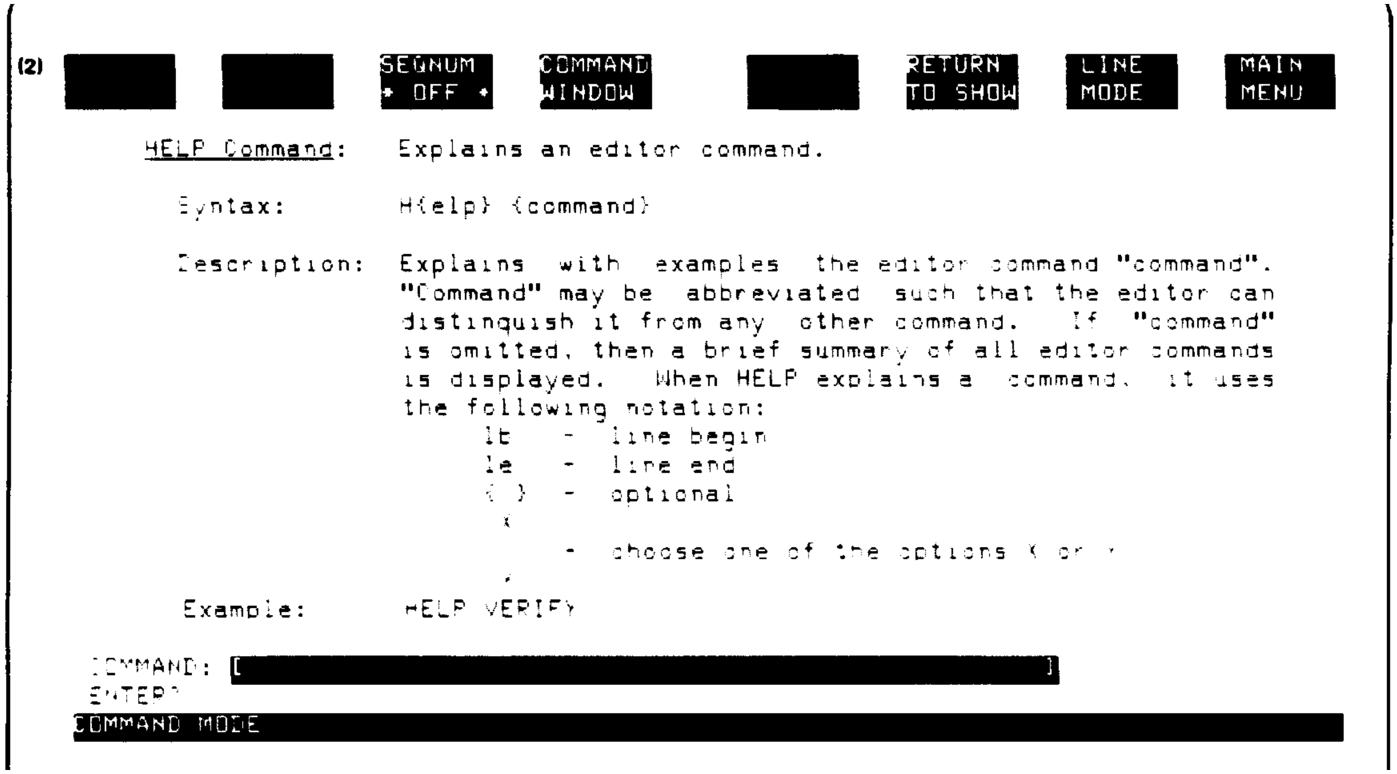
H A brief explanation of all RISE commands will appear on the screen.

The first example shows the effects of the HELP command when the "command" parameter is not specified. Note that the portions of individual commands printed in capital letters constitute the minimum abbreviations of those commands that RISE will accept (1).


(1)

```
                          SUMMARY OF COMMANDS:

Add      - add new lines.                  List     - list lines.
Begin    - start a new session.            MEnu     - display main menu(block mode).
Change   - change a string.                Modify   - modify lines.
COMment  - display comment banner.         MOVe     - move lines to new location.
COPy     - duplicate lines.                Print    - print lines offline.
Delete   - delete lines.                   Renum    - re-number sequence numbers or
Exit     - end system.                                source line numbers.
File     - edit file directly.             RUn      - run a program.
Find     - locate a string.                Show     - display a page in block mode.
FOrm     - display RPG Spec form.          Text     - edit a copy of a file.
Get      - execute commands in a file.     Undel    - un-delete lines deleted by the
Help     - explain a command.                         last Delete command.
Incr     - set default increment.          Verify   - RPG compile or PREP USL or
Join     - append or merge a file.                    LIST the compilation listing.
Keep     - save the edit file.             Xpand    - expand the edit file size.
Line     - enter line mode.                :        - execute MPE command.
 COMMAND: [                                                    ]
 ENTER
COMMAND MODE
```

The second example shows a typical detailed explanation of a command--in this case the HELP command itself (2). (The command which brought the second example to the screen, if abbreviated to the minimum, could have been "H H".)


(2)

```
   HELP Command:   Explains an editor command.

     Syntax:       H{elp} {command}

     Description:  Explains with examples the editor command "command".
                   "Command" may be abbreviated such that the editor can
                   distinguish it from any other command.  If "command"
                   is omitted, then a brief summary of all editor commands
                   is displayed.  When HELP explains a command, it uses
                   the following notation:
                         lb   -  line begin
                         le   -  line end
                         { }  -  optional
                          {
                              -  choose one of the options { or }
                          }
     Example:      HELP VERIFY

 COMMAND: [                                                  ]
 ENTER
COMMAND MODE
```

INCR

INCRement resets default increment.

Form

```
     I[NCR] [value]
```

Parameter Explanation

`value          New increment value you wish to establish.  (For limitations, see Note below.)`

Purpose

The INCRement command resets the default INCRement to your new "value". Once you use the INCR command, all subsequent commands which require the increment value will use your new value unless you specify a different increment in the parameters of the command. Typing "I[NCR]" without "value" resets the default value of RISE, which is 1.

NOTE There are limits to the INCREMENT value that RISE will accept. The increment value must be between 0.001 and 5000 inclusively.

Execution mode: Line or Block

Examples

The following are legal abbreviations which may be used with the INCrement command and its parameters:

```
I 100            Sets increment value to 100.

I                Resets increment value to default system value of 1.
```

In the first example, an error message results from an incorrect entry of the INCR command (1). When a parameter that should be numeric is entered alphabetically, a second error message appears (2). The command is then entered correctly but a listing of the file in this case a five line truncation of SIMCAL--does not indicate any change in the increment value (3). However, when RENUMber is entered, a LIST ALL then produces the effects of the changed increment value (4).

```
     (1) >INCC
             ^
          Error 21: Syntax error on INCR command.

     (2) >INCR FIVE
                ^
          Error 76: Illegal numeric increment value.
         >INCR 5
          Increment set.

     (3) >L A
             1     00011H
             SIMCAL
             2     00012FINPUT   I P  F      72                  DISK
             3     00013FOUTPUT  O    F      72                  DISK
             4     00014IINPUT   AA   01  1 CA
             5     00015I       OR    02  1 CS
         >RENUM ALL
          Renum completed.
```

```
    >L ALL
        5     00011H
        SIMCAL
       10     00012FINPUT   I P  F     72                  DISK
       15     00013FOUTPUT  O    F     72                  DISK
       20     00014IINPUT   AA   01  1 CA

(4)    25     00015I       OR    02  1 CS
```

In the second example, the I command without a value resets the default increment value of 1, and the RENUMber command is entered (1). When an illegal abbreviation of LIST is entered, an error is generated (2). A legal abbreviation of LIST ALL is then entered, and the command is executed (3).

```
     >I
(1)   Increment reset to 1.
     >RENUM
      Renum completed.

(2)  >LI AL
        ^
      Error 112: Can't distinguish between LINe or LISt command.

(3)  >LIS AL
         1     00011H
         SIMCAL
         2     00012FINPUT    IP  F       72          DISK
         3     00013FOUTPUT   O   F       72          DISK
         4     00014IINPUT    AA  01   1  CA
         5     00015I        OR   02   1  CS
```

In the third example, a series of errors establishes the limits of increment values acceptable to RISE. It will not accept a value of zero (1). The minimum increment cannot be be more than three digits to the right of the decimal point (2). The maximum cannot exceed 5000 (3).

```
(1)  >I0
      Error 78: An increment value cannot be zero.

(2)  >I.0001
            ^
      Error 74: A maximum of 3 digits is allowed right of decimal.

(3)  >IN6000
      Error 156: BY increment cannot be greater than 5000.
     >
     >
```

JOIN

JOIN merges or appends a file to your work file.

Form

```
J[OIN] filename [TO loc] [BY inc]
```

Parameter Explanation

filename — Name of the file to be joined.

TO loc — Location where "filename" will be joined. If the destination line exists, "filename" will be joined at the next line in sequence.

BY inc — Increment value of sequence number you wish to use for the JOIN command.

Purpose

The JOIN command appends or merges the entire contents of file "file name" to the work file. If "TO loc" is not specified "filename" is appended at the end of the work file. If "BY inc" is not specified, the current increment default value is used.

Using an increment value of less than 1 allows you to concentrate more lines in a restricted location. Sometimes the entire contents of "filename" cannot be joined as indicated because they are blocked by the next higher line number. RISE will automatically attempt to join them by using the smallest increment value of 0.001. If this attempt fails because of the block, the JOIN command will not execute.

```
 *  Execution mode:  Line or Block.

 *  Record Pointer:  At last line joined to the work file.

 *  Control Y:    Stops listing of lines being joined to work file.
```

Examples

The following are legal abbreviations which may be used with the JOIN command and its parameters:

```
J FILE2 TO 10    Join the file named FILE2 to the current work file starting at line 10.

J FILE2 TO L     Join the file named FILE2 to the current work file
                 starting at the first line following the last line in the file.

JOI FILE2 BY     Join the file named FILE2 at the end of the current work
0.1              file using an increment value of 0.1.
```

In the JOIN examples, a file named RPGDOC is created for purposes of file documentation. It is stored as a permanent file using the KEEP command (1). A five-line version of SIMCAL is then copied into a work file using the TEXT command (2). The JOIN command is used to join RPGDOC to SIMCAL at line 0.1 (3). The results are displayed on the screen (4), and an execution message appears (5).

The LIST command, entered incorrectly, produces an error message (6). When LIST is entered legally, with a line range from the first line to line 5 (7), the display indicates how the JOIN command with an increment of 0.1 is executed (8). The user could JOIN RPGDOC repeatedly to different source files to document those files.

```
                0        1        2        3        4
        1234567890123456789012345678901234567890123456789
           1    :    H*
           2    :    H*  PROGRAM NAME:
           3    :    H*
           4    :    H*  PROGRAMMER  :
           5    :    H*  DATE WRITTEN:
           6    :    H*  PURPOSE     :
           7    :    H*
           8    :<Control Y>
    (1) >K RPGDOC
         Keep completed.
         File name is RPGDOC
        >
        >
        >
        >T SIMCAL
         Text completed.
    (2) >L 1/5
           1      00011H
           SIMCAL
           2      00012FINPUT    I P  F      72              DISK
           3      00013FOUTPUT   O    F      72              DISK
           4      00014IINPUT    AA   01   1 CA
           5      00015I        OR    02   1 CS
```

```
(3)  >JOIN RPGDOC TO .1
          .1        H*
          .2        H*  PROGRAM NAME:
          .3        H*
          .4        H*  PROGRAMMER  :
          .5        H*  DATE WRITTEN:
          .6        H*  PURPOSE     :
          .7        H*

(4)   Join completed.

(5)  >

(6)  >LF/5
       ^
      Error 112: Can't distinguish between LINe or LISt command.

(7)  >L F/5
          .1        H*
          .2        H*  PROGRAM NAME:
          .3        H*
          .4        H*  PROGRAMMER  :
          .5        H*  DATE WRITTEN:
          .6        H*  PURPOSE     :
          .7        H*
         1     00011H
         SIMCAL
         2     00012FINPUT   IP   F       72           DISK
         3     00013FOUTPUT  O    F       72           DISK
         4     00014IINPUT   AA   01   1  CA

(8)      5     00015I       OR    02   1  CS
     >
```

KEEP

KEEP saves the contents of the work file in a permanent file.

Form

```
K[EEP] [filename] [N[OW]] [U[NN]]
```

Parameter Explanation

filename — Name of the permanent file where you wish to copy the work file.

N[OW] — Execute the command immediately, disabling safety prompts and overwriting an old permanent file with the same name if it exists. (Note: "filename" cannot be "N", "NO", "NOW", "U", "UN", or "UNN". These letters are reserved for literal parameters in the command.)

U[NN] — KEEP the file unnumbered (without the appended sequence numbers). If you omit "UNN", the sequence numbers will be stored in columns 81-88 of the permanent file.

Purpose

The KEEP command saves the contents of the work file in a permanent file designated "filename". If "filename" is omitted, the file will be kept under the current work file name which was designated previously with the TEXT command. If "filename" does not exist, RISE will create that file.

If "filename" exists, RISE will not immediately overwrite the existing file unless you use the NOW parameter. Instead, it will return a message informing you that "filename" already exists, and asking if you wish to overwrite the existing permanent file. You respond with "Y" to replace "N[O]" to cancel the KEEP command.

The "NOW" option ensures that the overwriting will take place immediately. It also disables the safety prompts. It saves time but it should only be used when you are certain that the work file should replace the permanent file. There is no provision for reconsidering changes once NOW is used. When you omit NOW, and respond to the message informing you that the "filename" exists and asking if you wish to overwrite the existing copy, you do have an opportunity to reconsider the replacement. However, there is a slight delay involved.

Related Commands

A file which you KEEP may be copied back to the work file using the TEXT command so that you can edit or add to it. To TEXT and KEEP a large RPG source file each time you have changes to make can entail long delays. You will save time by editing a long file directly using the FILE/EXIT commands. See the discussion of FILE for advantages and disadvantages of using the FILE/EXIT commands.

Execution mode: Line or Block

Examples

The following are legal abbreviations which may be used with the KEEP command and its parameters:

```
K FILE1 N        Copy the file named FILE1 into a permanent file.  If
                 FILE1 exists, overwrite the old copy with the contents
                 of the current work file without displaying safety prompts.

K NEWFILE        Copy the file named NEWFILE into a permanent file.

K N U            Copy the current work file into a permanent file
                 immediately without appended sequence numbers.
```

In the first example, SIMCAL is copied into the work file with the TEXT command. It is then copied to a permanent file called BACKUP using the KEEP command, and the execution messages are displayed (1). When lines 1 through 5 of the work file are deleted, and an attempt to KEEP BACKUP UNNumbered is entered, safety prompts appear (2), giving the user a chance to reconsider the effects of overwriting the original copy of BACKUP with the edited copy now in the work file. When the prompt is answered in the affirmative, the KEEP command is executed (3). Line 38 is then modified, and KEEP BACKUP is entered with the NOW and UNN parameters (4). NOW disables the safety prompts, and the command is executed immediately (5).

```
          >T SIMCAL
           Text  completed.
          >KEEP BACKUP
           Keep  completed.
     (1)   File name is BACKUP
          >
          >DELETE1/5
              1       00011 H
              SIMCAL
              2       00012FINPUT    IP  F       72                DISK
              3       00013FOUTPUT   O   F       72                DISK
              4       00014IINPUT    AA  01   1  CA
              5       00015I        OR   02   1  CS
           Delete  completed.

     (2)  >KEEP BACKUP UNN
           File  exist already, purge old BACKUP?YES
           Keep  completed.

     (3)   File  name is BACKUP
          >
          >MODIFY LAST
             38     00048O*END OF PROGRAM
                   :                      RNAMED SIMCAL
             38     00048O*END OF PROGRAM NAMED SIMCAL
                   :

     (4)  >KEEP BACKUP NOW UNN
           Keep completed.

     (5)  >File name is BACKUP
          >
```

In the second example, a file named TESTFILE is copied into the work file, TESTFILE was previously stored as a permanent file with the UNNumbered parameter. Because sequence numbers are essential as references in editing, they are appended automatically for the user's convenience (1). The CHANGE command is used on TESTFILE, and an execution message informs the user that all the changes have been made (2).

Entering EXIT causes safety prompts to appear because, if the EXIT is executed without a preceding KEEP command, the changes made to the work file would not be saved. When the prompts are answered negatively, the command is cancelled (3). KEEP is then tried, and the safety prompts displayed (4). This time the user indicates that the permanent copy of TESTFILE should be overwritten by the edited work file. The user enters the EXIT command, and this time it is safe to exit since the changes were kept in a permanent file first (5).

```
          >
          >TEXT TESTFILE
     (1)   File is unnumbered, appending sequence numbers.
           Text completed.
          >CHANGE ;RESULT; TO ;OUTPUT; IN ALL
              6     00021O   01      OPRND1    ADD  OPRND2    OUTPUT 104
              7     00022O   02      OPRND1    SUB  OPRND2    OUTPUT
              8     00023O   03      OPRND1    MULT OPRND2    OUTPUT
              9     00024O   04N99   OPRND1    DIV  OPRND2    OUTPUT   H
             10     00025O   04 99             Z-ADD0         OUTPUT
             23     00038O                           OUTPUT    60 "    .0 .
  -"

     (2)   Changes completed.
          >EXIT
           KEEP not done. CLEAR current work file?NO

     (3)   Exit cancelled.
          >KEEP
           File exist already, purge old TESTFILE?YES

     (4)   Keep completed.
           File name is TESTFILE

     (5)  >EXIT
```

LINE

LINE places you in Line Mode.

Form

```
     LIN[E]
```

Purpose

This command places you in Line Mode. There are many advantages to using LINE Mode, among them convenience and speed. The LINE command allows you to enter commands one at a time by pressing the carriage return. Errors can be erased with the backspace key.

In Line Mode you see the results of commands being executed on the screen. For example, if you use the

DELETE command, the lines being deleted are displayed. (You may not wish to have all the lines being deleted displayed. By typing <Control Y> you stop the listing. <Control Y> does not stop the execution of the command, however.)

Related Commands

Another method of entering Line Mode from Block Mode is the the special function key F7, when the softkey label for F7 reads "LINE MODE".

Execution mode: Line

Examples

The following is a legal abbreviation of the LINE command:

```
LIN             Shift to Line Mode.
```

LIST

LIST prints the work file on the screen or on the line printer.

Form

```
        L[IST] [lb[/le]] [O[FF]]
```

Parameter explanation

lb First or only line you want to have displayed.

le Last line of a range you want displayed on the screen.

O[FF] Print offline on a line printer with the formal file name of "RISELIST" and device class of "LP".

Purpose

The LIST command prints all or a portion of the work file on the screen or line printer. If you do not specify a line or lines for listing, only the line following the current position of the record pointer is listed. You specify the "OFF" parameter to list offline to a line printer.

Related Command

The PRINT command will also list lines offline on a line printer. PRINT has an additional feature: whenever the RPG form type in column 6 changes, it skips three lines so that your listing is in a more readable form. The PRINT command will also print offline the compilation listing produced by a V[ERIFY] R[PG] command.

```
 *  Execution mode:  Line.

 *  Record pointer:  At the last line printed.

 *  Control Y: Halts execution of the command.
```

EXAMPLES

The following are legal abbreviations which may be used with the LIST command and its parameters:

LIS A List the complete contents of the work file.

L F/10 List the first through line 10.

L *-5/*+5 List a range of lines beginning five lines before the current position of the record pointer and ending five lines after the record pointer.

L10 List line 10.

L List the line following the position of the current record pointer. (Used for stepping through a file, line by line.)

In the first example, TEXT is used to copy SIMCAL into a work file, and the first five lines are listed on the screen using the LIST command (1). The LIST command is then used with the asterisk--(*)--which symbolizes the current position of the record pointer. The command calls for a range of ten lines beginning five lines before the current record pointer position. Since the previous command left the record pointer at line 5, when the command is executed, lines 1 through 10 are displayed (2).

When the L[IST] command is used without any parameters, the line following the current position of the record pointer is listed. Thus, lines 11 through 13 are stepped through (3).

```
(1)  >T SIMCAL
      Text completed.
     >L 1/5
         1     00011H
         SIMCAL
         2     00012FINPUT    IP  F       72            DISK
         3     00013FOUTPUT   O   F       72            DISK
         4     00014IINPUT    AA  01   1  CA
         5     00015I        OR   02   1  CS

(2)  >L*-5/*+5
         1     00011H
         SIMCAL
         2     00012FINPUT    1P  F       72            DISK
         3     00013FOUTPUT   O   F       72            DISK
         4     00014IINPUT    AA  01   1  CA
         5     00015I        OR   02   1  CS
         6     00016I        OR   03   1  CM
         7     00017I        OR   04   1  CD
         8     000181        OR   05
         9     00019I                        8   72OPRND1
        10     00020I                        9  132OPRND2         99

(3)     >L
          11      00021C   01       OPRND1     SUB  OPRND2    RESULT  104

        >L
          12      00022C   02       OPRND1     SUB  OPRND2    RESULT
        >L
          13      00023C   03       OPRND1     MULT OPRND2    RESULT
```

In the second example, LIST is used to bring the last line of the file to the screen (1). Next, the special character "@", which has the same meaning to RISE as the asterisk (*), is used with LIST to display the five lines preceding the current position of the record pointer--at line 38 after the last command. Thus, the last five lines of the file are listed (2).

```
(1)  >L LAST
        38      00048O*END OF PROGRAM

(2)  >L @-5/L
        33      00043O        D 11      1P
        34      00044O       OR        N1P
        35      00045O                           13 "C NNNDD NNNDD"
        36      00046O        D         05
        37      00047O                           15 "INVALID REQUEST"
        38      00048O*END OF PROGRAM
```

In the next example, an error results when non-existent lines are used with the LIST command (1). Following that, LIST is entered with a "line beginning" which is higher in sequence than the "line ending", and another error message is shown (2). Next, the same error is committed using special characters to indicate the current position of the record pointer (3). LIST used with a line number which includes an illegal decimal value causes another error (4). Finally, an illegal character used with the FIRST parameter brings up another error message.

```
(1)  >L 100/150
      Error 81: Range is empty.

(2)  >L 10/5
      Error 62: The TO line cannot be less than the FROM line number.

(3)  >L @+5/@-5
      Error 62: The TO line cannot be less than the FROM line number.

(4)  >L 10.0005
               ^
      Error 74: A maximum of 3 digits is allowed right of decimal.
     >

(5)  >L FIRST/
      Error 61: Only a '*',line number,FIRST,LAST,ALL allowed in range.
     >
```

MENU

MENU displays the main menu.

Form

```
ME[NU]
```

Purpose

The MENU command displays the main menu. It contains four sets of commands and the labels of the special function keys. The command sets are segregated by function: File Commands, Editing Commands, General-1 Commands, and General-2 Commands. See the example of the menu below for the commands which are associated with each heading.

Related commands

To enter a command, you press the special function key which corresponds to the set containing the commands you wish to use. The softkey moves the cursor to the first inverse video (white) box. You type in the command abbreviation. The cursor then moves to the next box, where you enter the parameters of the command. Under each box is a description of the appropriate parameter for that space. After the menu is filled out, you press the ENTER key to execute the command.

NOTE Note that not all RISE commands appear on the menu. Only the most important do.

When the main menu is displayed, the special function keys perform the following operations:

F1: Move cursor to File Commands.

F2: Move cursor to General-1 Commands.

F3: Show RPG source lines with or without editor sequence numbers when you enter the next

SHOW command.

F4: Move cursor to Command Window.

F5: Move cursor to Edit Commands.

F6: Move cursor to General-2 Commands.

F7: Enter Line Mode.

F8: Refresh main menu (clears the screen and redraws the menu).

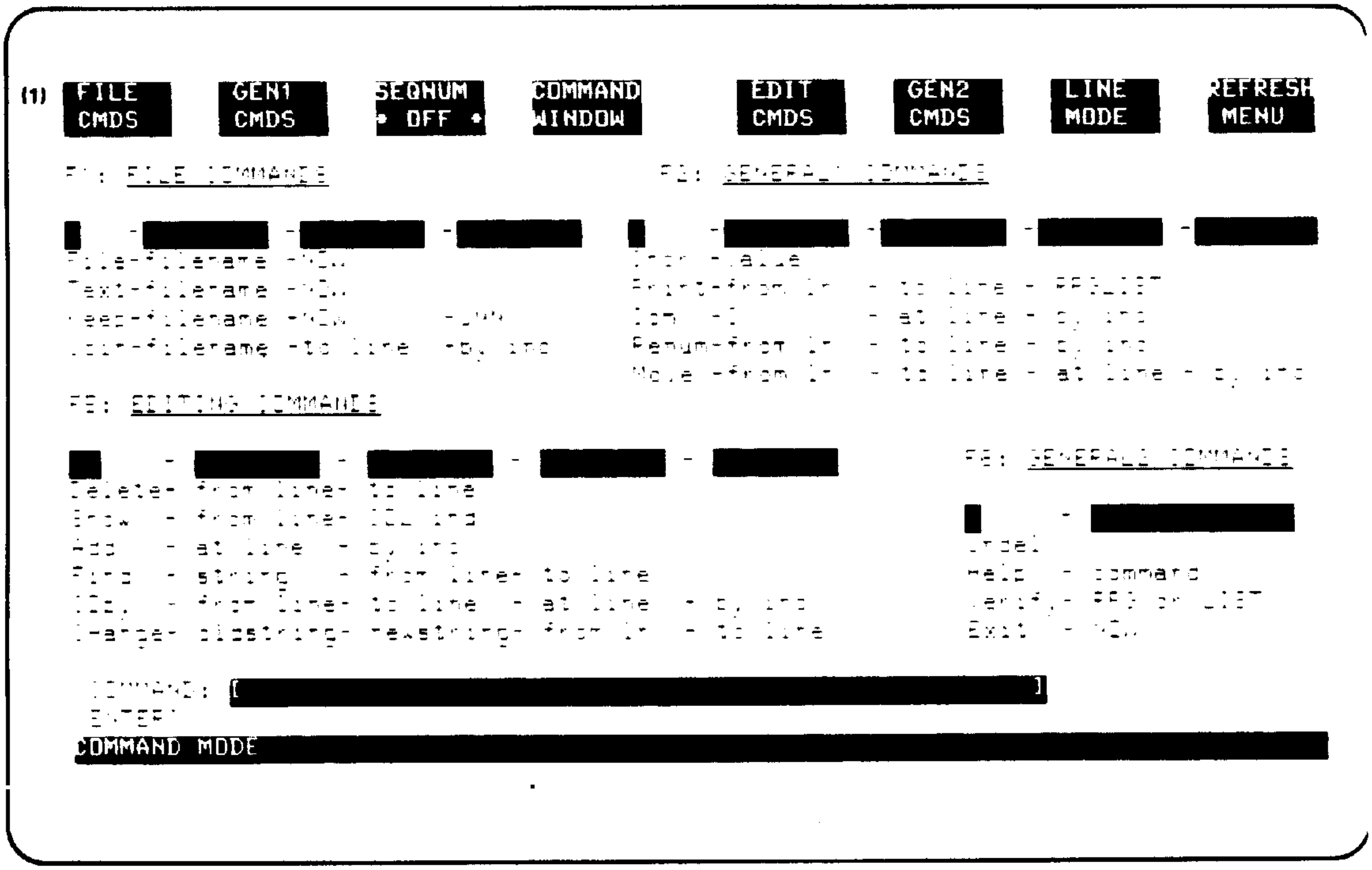
Execution mode: Block.

Examples

The following is a legal abbreviation which may be used for the MENU command:

ME Display the main command menu.

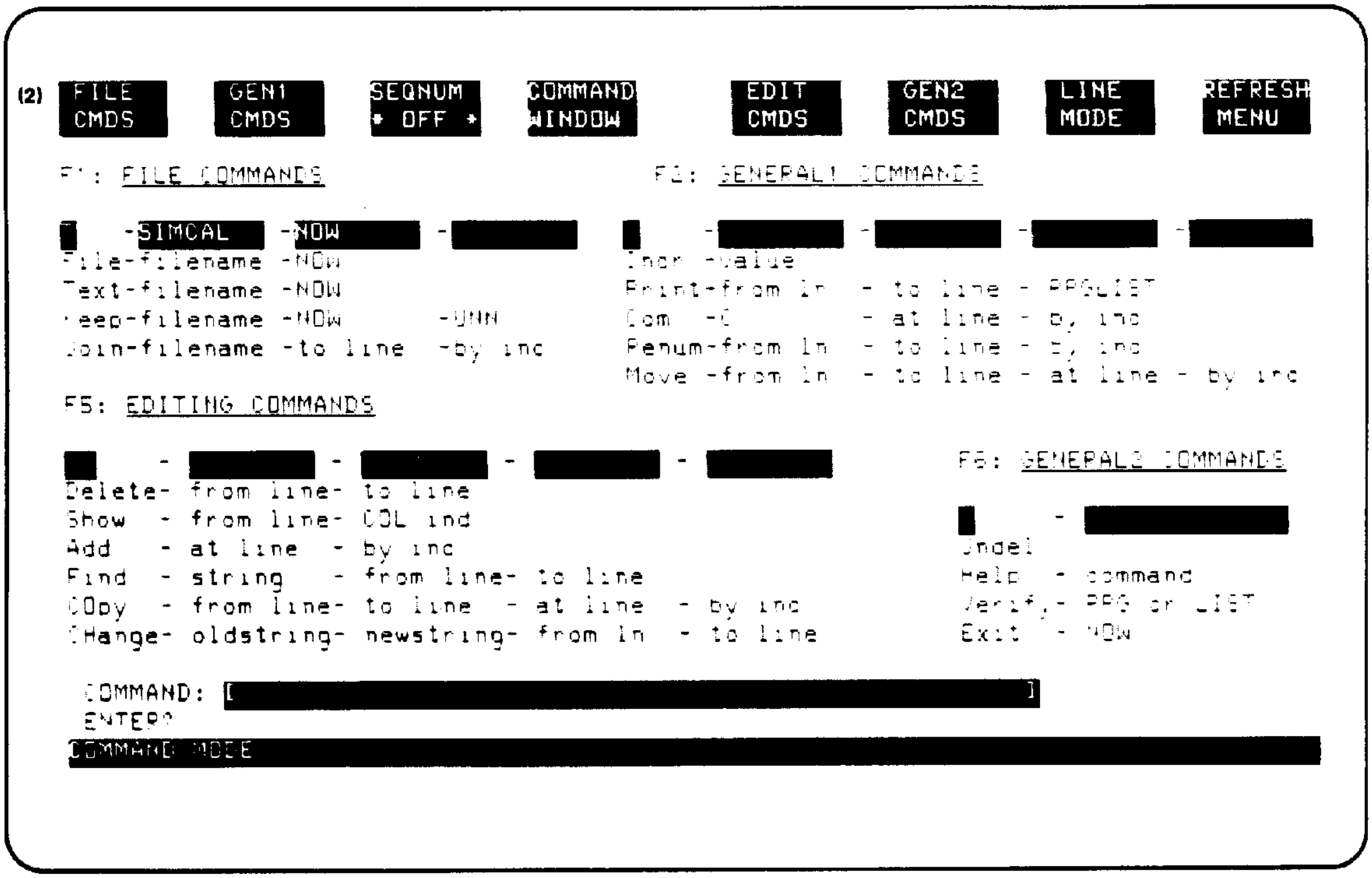
The first example shows the main menu without any commands entered in any of the unprotected fields (1).



```
F1: FILE COMMANDS                         F2: GENERAL1 COMMANDS

File-filename -NEW                        Incr -value
Text-filename -NEW                        Print-from ln  - to line - PRLIST
Keep-filename -NEW      -DUP              Ins  -1        - at line - by inc
Join-filename -to line  -by inc           Renum-from ln  - to line - by inc
                                          Move -from ln  - to line - at line - by inc
F5: EDITING COMMANDS

                                                              F6: GENERAL2 COMMANDS
Delete- from line- to line
Show  - from line- ALL inc
Add   - at line  - by inc                                     Undel
Find  - string   - from line- to line                         Help  - command
Copy  - from line- to line  - at line  - by inc               Verify- PRT or LIST
Change- oldstring- newstring- from ln  - to line              Exit  - NEW

 COMMAND: [                                              ]
 ENTER
COMMAND MODE
```

The second example shows the menu with the TEXT command selected in the upper left corner. Two parameters are also entered to complete the full command of "TEXT SIMCAL NOW" (2).

(2)



```
FILE     GEN1     SEQNUM    COMMAND      EDIT     GEN2     LINE     REFRESH
CMDS     CMDS     * OFF *   WINDOW       CMDS     CMDS     MODE     MENU

F1: FILE COMMANDS                      F2: GENERAL1 COMMANDS

    -SIMCAL    -NOW       -                 -         -         -         -
File-filename -NOW                     Incr -value
Text-filename -NOW                     Print-from ln  - to line - RPGLIST
Keep-filename -NOW        -UNN         Com  -C        - at line - by inc
Join-filename -to line   -by inc       Renum-from ln  - to line - by inc
                                       Move -from ln  - to line - at line - by inc
F5: EDITING COMMANDS

     -          -          -          -                  F6: GENERAL2 COMMANDS
Delete- from line- to line
Show  - from line- COL ind                                   -
Add   - at line  - by inc                                Undel
Find  - string   - from line- to line                    Help  - command
COpy  - from line- to line  - at line  - by inc          Verify- RPG or LIST
CHange- oldstring- newstring- from ln  - to line         Exit  - NOW

 COMMAND: [                                         ]
 ENTER?
COMMAND MODE
```

MODIFY

MODIFY enables you to modify lines interactively.

Form

```
M[ODIFY] [lb[/le]] [^^]
```

Parameter Explanation

lb	Single line or first line of range you wish to modify.
le	Last line of range you intend to modify.
^^	Displays the line or lines for modification with the corresponding RPG Record Specification form.

Purpose

The MODIFY command is your means of modifying lines interactively. It operates similarly to the same command in EDIT/3000. The line which is selected for modification is displayed at the terminal. You modify the line by positioning the cursor beneath the first, or only, character you wish to modify Three subcommands operate with MODIFY. They are insert (I), replace (R), and delete (D).

Related Commands

Insert ("I") Subcommand

"Istring" will insert a character or string. You position the cursor beneath the position where you wish the insertion to begin. You type an "I" immediately followed by the "string" to be inserted.

Replace ("R") Subcommand

"Rstring" will replace the characters in the line marked "R" by your new "string". Again, you position the cursor beneath the first character you wish to replace before typing "Rstring".

Delete ("D") Subcommand

To delete a character or string of characters, type "D" beneath the first character to be deleted. A range of "D"'s will remove the string between the first and last "D". If there is more than one "D" used in modifying a single line, all characters between the leftmost and rightmost "D" will be deleted.

You can use "I" (insert) and "R" (replace) with a "D" or range of "D"'s. If only one "D" is used, followed by an "I" or "R", the second "D" is inferred to be the last character before the "I" or "R". If more than one "D" is used, the characters between the first and last will be deleted, and the second subcommand will be executed as usual at the character marked by the "I" or "R".

Undesignated Subcommand

When you use the MODIFY command, the line designated appears on the screen. If you do not indicate the subcommand you are using (D, I, or R), whatever is typed beneath the line to be modified will replace the character or characters directly above unless blanks alone are typed. In other words, R(eplace) is the default subcommand.

Leaving MODIFY

If you specify a range of lines to be modified and wish to leave the mode before going through the full range, type double slash (//) at the beginning of the next line. When you are modifying a single line, press the carriage return at the beginning of a new line after the source line appears on the screen in the form desired.

Control Y

If you do not wish to make the modifications you originally thought were necessary, type <Control Y> before you press the carriage return at the start of a new line. (Pressing the carriage return at the start of a new line means leaving modification mode for this line and permanently storing the newly modified line.) RISE will respond with the message "Restore Record", and the changes will not be made.

The UNDELete command does not work with deletions made in the the MODIFY mode.

```
 *  Execution mode:  Line.

 *  Record pointer:  At last line modified.
```

Examples

The following are legal abbreviations which may be used with the MODIFY command and its parameters:

M */LAST^^	Modify lines from the current position of the record pointer to the last line and display the corresponding RPG Record Specification form.
M LAST	Modify the last line of the file.
M ^^	Modify line at the current position of the record pointer and display the corresponding RPG Record Specification form.

M Modify line at the current position of the record pointer.

MOD AL^^ Modify all the lines and display the corresponding RPG Record Specification form.

The first five lines of SIMCAL are used for these examples. After the lines are listed, MODIFY FIRST is entered, and a two-character string is replaced using the R subcommand (1). Note that the line selected for modification is displayed unmodified on the screen initially, and that it is displayed again after the execution of the subcommand. An effort to use an illegal parameter with MODIFY evokes an error, and the suitable message is displayed (2).

Line 2 is singled out for a number of modifications. Initially, the word INPUT is deleted. Note that the D's are placed directly beneath the first and last characters of the string INPUT, and that this is sufficient to delete the complete word (3). <Control Y> is used to demonstrate a method of restoring an erroneous deletion (4).

```
     >L 1/5
         1      00011H
         SIMCAL
         2      00012FINPUT    IP  F       72               DISK
         3      00013FOUTPUT   O   F       72               DISK
         4      00014IINPUT    AA  01   1  CA
         5      000151        OR   02   1  CS
(1)  >MOD FIRST
         1      00011H
         SIMCAL
              :    R10
         1      00010H
         SIMCAL
              :

(2)  >M SECOND
        ^
      Error 61: Only a '*',line number,FIRST,LAST,ALL allowed in range.

(3)  >M 2
         2      00012FINPUT   IP  F     72            DISK
               :     D    D

(4)      2      00012F   IP  F      72           DISK
               :<Control Y>  Restore Record.
```

In the next example, the delete subcommand removes a three-character string from the word INPUT (1). Note that the remainder of the line moves left a corresponding number of spaces (2). Thus, the syntax of an RPG source program will be affected by modifications which remove characters or insert characters into records. Next, the string removed by the delete subcommand is replaced by a three character insertion (3).

The replace (R) subcommand is used next (4). Though two more characters are added to the line, they replace blank spaces, and the syntax of the line is not changed. Replace (R) is used again on the next line (5). And the word DISK is replaced by TAPE in the line following, although the subcommand is not used. This is an example of an undesignated subcommand (6).

```
(1)   2      00012FINPUT    IP  F         72                DISK
            :        D D

(2)   2      00012FIN   IP F            72                DISK

(3)         :        IFIL

(4)   2      00012FINFIL    IP  F         72                DISK
            :           RES

(5)   2      00012FINFILES  IP  F         72                DISK
            :                             R80
```

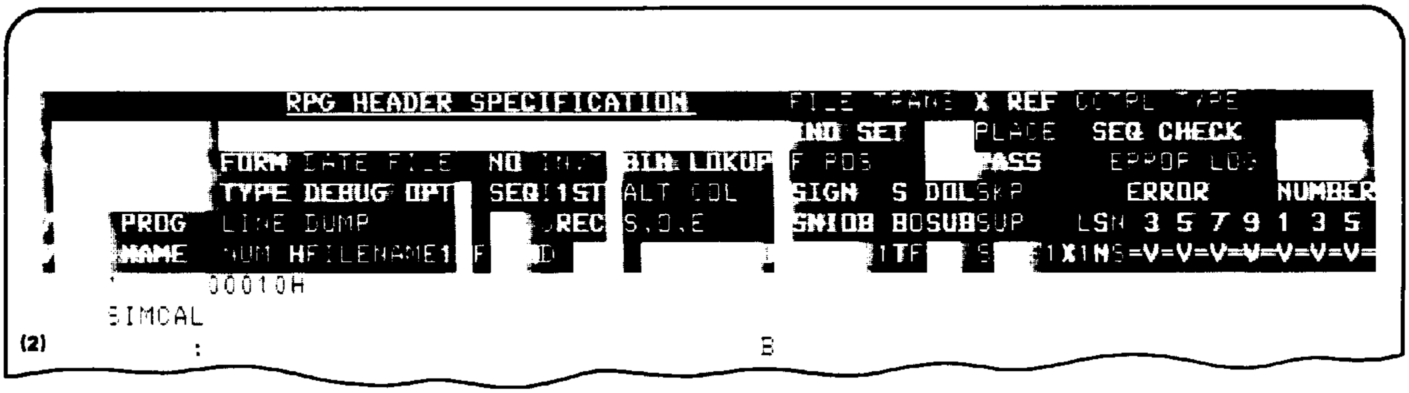
```
(6)   2      00012FINFILES  IP  F         80                DISK
            :                                               TAPE
```

In the next example, the delete (D) subcommand is used in conjunction with the insert (1) subcommand with the result that "MAGTAPE" replaces "TAPE" in the line (1). The next example demonstrates how all characters between the two outside D's used in the delete subcommand are removed upon execution of the subcommand (2). Again a <Control Y> is used to restore the characters erroneously deleted (3). Finally, the delete and insert subcommands are again used together, and the results are shown (4).

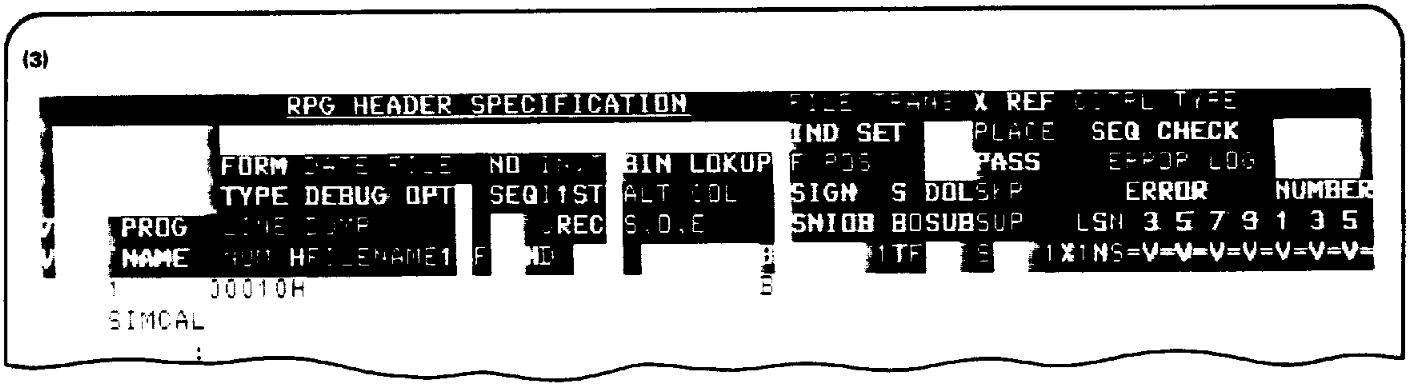
```
(1)   2      00012FINFILES  IP  F         80                TAPE
            :                                               D  DIMAGTAPE
(2)   2      00012FINFILES  IP  F         80                MAGTAPE
            :D DDD                       D

(3)   2      80             MAGTAPE
            :<Control Y>   Restore Record.

(4)   2      00012FINPUT    IP  F        72                 DISK
            :DDDDDI12345
      2      12345FINPUT    IP  F        72                 DISK
            :
     >
```

In the final example of the MODIFY command, MODIFY ALL is entered with the additional parameter calling for the record specification form corresponding to the line being modified (1). The first line, then, is shown beneath the Header Specification form (2). It is modified by spacing over and typing a "B" as shown under the form (2). After typing the "B", the user hits the carriage return, and RISE re-displays the modified line (3). Lines 2 and 3 are then displayed under the RPG Specification form (4) (5). A double slash (//) entered at the beginning of line 3 ends the modification mode (6).

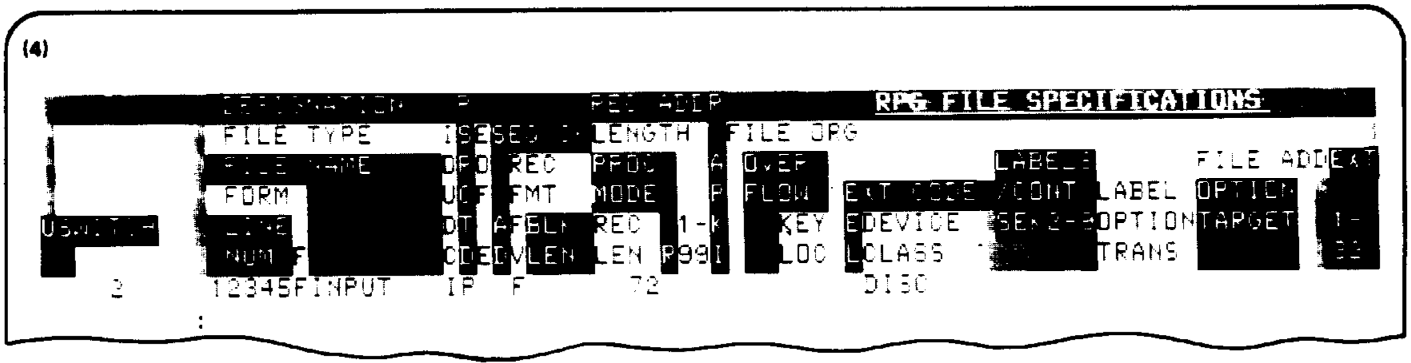
```
 (1) >MODIFY ALL ^^
```



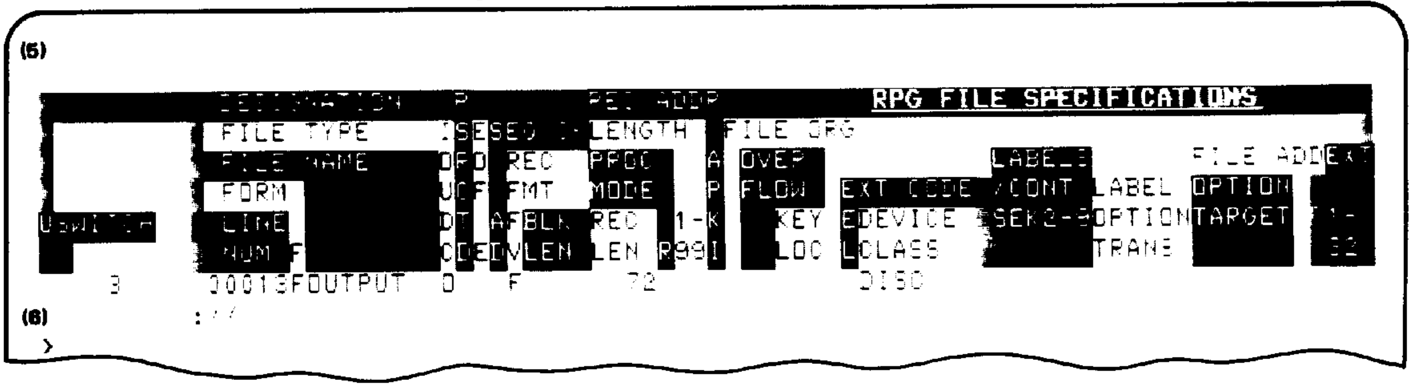
```
          RPG HEADER SPECIFICATION      FILE TRANS X REF CCTRL TYPE
                                        IND SET      PLACE  SEQ CHECK
          FORM DATE FILE  NO INPT BIN LOKUP F POS    PASS   ERROR LOG
          TYPE DEBUG OPT  SEQ 1ST ALT COL   SIGN  S DOLSKP    ERROR     NUMBER
    PROG  LINE DUMP           REC S,D,E     SNIOB BOSUBSUP  LSN 3 5 7 9 1 3 5
    NAME  NUM HFILENAME1 F  D                    1TF  S  1X1NS=V=V=V=V=V=V=V=
          00010H
    SIMCAL
(2)     :                                 B
```



```
(3)

          RPG HEADER SPECIFICATION      FILE TRANS X REF CCTRL TYPE
                                        IND SET      PLACE  SEQ CHECK
          FORM DATE FILE  NO INPT BIN LOKUP F POS    PASS   ERROR LOG
          TYPE DEBUG OPT  SEQ 1ST ALT COL   SIGN  S DOLSKP    ERROR     NUMBER
    PROG  LINE DUMP           REC S,D,E     SNIOB BOSUBSUP  LSN 3 5 7 9 1 3 5
    NAME  NUM HFILENAME1 F  ND                   1TF  S  1X1NS=V=V=V=V=V=V=V=
    1     00010H                          B
    SIMCAL
        :
```



```
(4)

          DESIGNATION    P        REC ADDR            RPG FILE SPECIFICATIONS
          FILE TYPE      ISESEQ   LENGTH   FILE ORG
          FILE NAME      DFO REC  PROC   A OVER              LABELS        FILE ADDEXT
          FORM           UCF FMT  MODE   P FLOW    EXT CODE /CONT LABEL OPTION
USWITCH   LINE           DT AFBLK REC 1-K   KEY   EDEVICE  SEK2-BOPTIONTARGET  1-
          NUM F          CDEDVLEN LEN R99I  LOC   LCLASS          TRANS        32
   2      12345FINPUT    IP  F      72             DISC
          :
```



```
(5)

          DESIGNATION    P        REC ADDR            RPG FILE SPECIFICATIONS
          FILE TYPE      ISESEQ   LENGTH   FILE ORG
          FILE NAME      DFO REC  PROC   A OVER              LABELS        FILE ADDEXT
          FORM           UCF FMT  MODE   P FLOW    EXT CODE /CONT LABEL OPTION
USWITCH   LINE           DT AFBLK REC 1-K   KEY   EDEVICE  SEK2-BOPTIONTARGET  1-
          NUM F          CDEDVLEN LEN R99I  LOC   LCLASS          TRANS        32
   3      00013FOUTPUT   O   F      72             DISC
(6)       ://
 >
```

MOVE

MOVE transfers lines to a new location.

Form

```
MOV[E] lb[/le] TO loc [BY inc]
```

Parameter Explanation

lb	Single line you wish to move or the beginning of a range of lines you are moving.

le	Last line of range you wish to move.

TO loc	Location where the lines are to appear. If destination line exists, the lines will be moved starting at the next line in sequence.

BY inc	Increment of line numbers you wish to use. If "BY inc" isn't specified, the current default increment is used If you attempt to move too many lines into a restricted position, the move may be blocked by the next higher line number. When this occurs, none of the lines are moved.

Purpose

The MOVE command transfers a line or lines to a designated location.

Related Commands

The COPY command duplicates a line or lines in a new position while leaving the original lines where they are. The MOVE command does not leave lines in their original position; they are physically transferred.

- Execution mode: Line or Block.
- Record pointer: At the line following the last line that was moved.
- Control Y: Stops the listing of lines being moved.

Examples

The following are legal abbreviations which may be used with the MOVE command and its parameters:

```
MOV 10 TO 20      Move line 10 to line 20.  (If data has been entered on
                  line 20, RISE will move 10 to the next higher line by
                  automatically creating an increment such as 20.1 if that is possible.)

MOV *-5/*+5 to    Move a range of lines beginning 5 lines before the
*+20              current record pointer to a position 20 lines after the
                  current record pointer.

MOV F TO LAST     Move the first line to the last line in the file.
```

In the first example, SIMCAL is copied into a work file using the TEXT command, and five lines are brought to the screen using LIST (1). The MOVE command is then used to move line 2 to line 3.5 (2). The results are automatically displayed (3). Lines 1 through 5 are then LISTed (4).

```
     >T SIMCAL
      Text completed.
(1)  >L 1/5
         1     00011H
         SIMCAL
         2     00012FINPUT    IP  F       72                DISK
         3     00013FOUTPUT   O   F       72                DISK
         4     00014IINPUT    AA  01   1  CA
         5     00015I        OR   02   1  CS

(2)  >MOVE 2 TO 3.5
         3.5  00012FINPUT     IP  F       72                DISK

(3)   Move completed.
```

```
(4)  >L1/5
         1     00011H
         SIMCAL
         2     00013FOUTPUT   O   F       72                DISK
         3.5   00012FINPUT    IP  F       72                DISK
         4     00014IINPUT    AA  01   1  CA
         5     00015I        OR   02   1  CS
```

In the next example, the record pointer is moved to the last line in the file when LIST LAST is entered (1). A command is then given to MOVE a range of lines beginning ten lines before the current record pointer and ending with the last line in the file to line 100 (2). The results appear on the screen (3). An attempt to move an empty line range brings up an error message (4). Another error is committed when a MOVE command is entered with only a portion of the line range specified (5). Finally, a third error is caused by a MOVE command used with an illegal destination value (6).

```
(1) >L LAST
       38     00048O*END OF POGRAM

(2) >MOVE *-10/LAST TO 100
      100     00038O                     RESULT  60 "   , 0 .    -"
      101     00039O    D 1        1P
      102     00040O   OR         N1P
      103     00041O                                 10 "ENTER DATA"
      104     00042O                                 25 "IN THE FORMAT:"
      105     00043O    D 11       1P
      106     00044O   OR         N1P
      107     00045O                                 13 "C NNNDD NNNDD"
      108     00046O    D          05
      109     00047O                                 15 "INVALID REQUEST"

(3)   110     00048O*END OF PROGRAM
     Move completed.

(4) >MOVE 50/70 TO FIRST BY .11
     Error 81: Range is empty.

(5) >MOVE FIRST
     Error 149: TO location missing.

(6) >MOVE FIRST TO .1155
                        ^
     Error 74: A maximum of 3 digits is allowed right of decimal.
    >
```

PRINT

PRINT enables you to print a complete work file or portion of the file, or a compilation listing of a file, on the line printer.

Form

```
P[RINT] lb[/le] [R[PGLIST]]
```

Parameter Explanation

lb First, or single, line you wish to have printed.

le Last line of a range you want printed offline.

RPGLIST Print the last RPG compilation listing produced by the RISE VERIFY command.

Purpose

The PRINT command prints a line, a line range, or a complete file on the line printer in a special format with the formal file name of "RISELIST" and device class of "LP". Whenever the RPG form type in column 6 changes, the PRINT command will skip three lines to group together those lines which have the same form type. This produces a more readable format for the source code. Because of this feature, the PRINT command should only be used when you are working on RPG source code. If "RPGLIST" is specified, the compilation listing produced by the VERIFY RPG command is printed instead of the work file. In this case, the formal file name used is "RPGLIST".

Related Commands

The LIST command will also print your workfile offline on a line printer. It does not skip 3 lines when column 6 changes. Thus, it is suitable for printing offline any type of text you may be editing, such as data or a document.

*Execution mode: Line or Block

Examples

The following are legal abbreviations which may be used with the PRINT command and its parameters:

```
P FIRST/50       Print a range of lines from the beginning to line 50.

P ALL RPGLIST    Print the entire listing produced by the VERIFY RPG command.

PRI A R          Print the entire listing produced by the VERIFY RPG command.

P*-5/L           Print a range of lines beginning five lines before the
                 current position of the record pointer and ending with
                 the last line in the file.
```

In the example, the TEXT command is used to copy SIMCAL into a work file. PRINT ALL is entered, and an execution message appears on the screen when the command is executed (1). An error results when PRINT is used with the RPGLIST paramater, and SIMCAL has not been previously compiled using the VERIFY RPG command (2). Finally, an abbreviation of the PRINT command is shown to be legal by the appearance of the execution message (3).

```
          >T SIMCAL
           Text completed.
     (1)  >PRINT ALL
           Print completed.
          >P ALL RPGLIST

     (2)   Error 64: There is no compilation listing to print.
          >
          >PRIN1/10

     (3)  Print completed.
          >
          >
```

RENUM

RENUM renumbers sequence line numbers.

Form

```
                   [lb/le]    [BYinc]
                   BOTH       [BYinc]
         R[ENUM]   SO[URCE]   [BYinc]
                   SO[URCE]   W[ITH]   SE[QNUM]
                   SE[QNUM]   W[ITH]   SO[URCE]
```

Parameter Explanation

Parameter option 1:[lb/le] [BY inc]

Renumbers the range "lb/le" by "inc". If the line range is omitted, the entire file will be renumbered. (As in other commands, "ALL" may be used in place of "lb/le".) If "BY inc" is omitted, the default increment is used.

Parameter option 2:BOTH [BY inc]

Renumbers both the editor sequence numbers and the RPG source line numbers (columns 1-5) by "inc". Both will have the same value. The value of "inc" cannot be less than 1. Using this option you renumber your entire file.

Parameter option 3: SO[URCE] [BY inc]

Renumbers the RPG Source program line numbers (col. 1-5). If "BY inc" is omitted, the line numbers are renumbered with the following bracketing values established for the record specification forms:

H starts at	10
F	6000
E	7000
L	8000
I	10000
C	30000
O	50000

If "BY inc" is specified, the source will be renumbered accordingly. Again, "inc" cannot be less than 1.

Parameter option 4: SO[URCE] W[ITH] SE[QUENCE]

The RPG source line numbers are renumbered to agree with the work file sequence numbers.

Parameter option 5: SE[QUENCE] W[ITH] SO[URCE]

The work file sequence numbers are renumbered to agree with the RPG source line numbers. If each line number in the RPG source program is not in ascending order, the RENUM command will not renumber the work file.

Purpose

The RENUM command renumbers sequence line numbers in the work file or line numbers in the RPG source program (columns 1-5).

Execution mode: Line or Block

Control Y: Stops the listing of lines that are being renumbered; does not stop the renumbering.

Examples

The following are legal abbreviations which may be used with the RENUMber command and its parameters:

```
R ALL           Renumber all sequence line numbers in the work file.

R BOTH BY 10    Renumber sequence line numbers and source line numbers
                using an increment value of 10.
```

```
R SO W SE         Renumber source line numbers so that they correspond
                  with sequence line numbers.

REN SO            Renumber source line numbers so that they have
                  bracketing values according to the record specification
                  form with which they are associated.
```

The examples use a truncated version of SIMCAL consisting of the first five lines. It has been stored in a permanent file with the name PARTSIM. When an attempt to TEXT the file is made, and the filename is typed incorrectly, an error message is returned (1). The command is then given correctly, and the file contents are listed (2). RENUM is entered along with a parameter calling for an increment value of 100.

When the command has been executed, RISE returns a message (3). After the LIST command is executed, the renumbered lines are shown on the screen (4). RENUM BOTH follows, with the BY 1 parameter. Another listing reveals how both sequence line numbers, and source line numbers are affected by this command (5).

```
(1)  >TEX PARTSIM2
       NONEXISTENT PERMANENT FILE (FSERR 52)

(2)  >TEX PARTSIM
      Text completed.
     >L ALL
         1     00011H
         SIMCAL
         2     00012FINPUT    IP  F       72               DISK
         3     00013FOUTPUT   O   F       72               DISK
         4     00014IINPUT    AA  01   1  CA
         5     00015I        OR   02   1  CS

(3)  >RENUM ALL BY 100
      Renum completed.

(4)  >L ALL
       100     00011H
         SIMCAL
       200     00012FINPUT    IP  F       72               DISK
       300     00013FOUTPUT   O   F       72               DISK
       400     00014IINPUT    AA  01   1  CA
       500     00015I        OR   02   1  CS
     >
     >RENUM BOTH BY 1
      Renum completed.
     >L ALL
         1     00001H
         SIMCAL
         2     00002FINPUT    IP  F       72               DISK
         3     00003FOUTPUT   O   F       72               DISK
         4     00004IINPUT    AA  01   1  CA

(5)      5     00005I        OR   02   1  CS
     >
```

In the next example, a misspelling of SOURCE evokes an error (1). When the command is entered correctly and listed on the screen, the source line numbers are seen to be numbered according to the bracketing values (2). Next, another typing error causes the display of an error message (3). The command, entered correctly, causes the sequence line numbers to take on the same numbers as the source line numbers (4). The source line numbers are then changed by a command which sets their increment value at 10. The results are brought to the screen by a LIST command (5).

An error results from a mistaken parameter following WITH (6), and a message is displayed. When a command to renumber source lines with sequence line numbers is correctly entered, and the LIST command is used, the results are displayed on the screen (7).

```
(1)  >RENUM SOUCE
               ^
      Error 161: Syntax error on SOURCE.
     >RENUM SOURCE
      Renum completed.
     >L FIR/L
         1     00010H
         SIMCAL
         2     06000FINPUT    IP  F       72               DISK
         3     06001FOUTPUT   O   F       72               DISK
         4     10000IINPUT    AA  01   1  CA
         5     10001I        OR  02   1  CS

(2)  >

(3)  >RENUM SEQNUM WITT SOURCE
                      ^
      Error 164: Syntax error on WITH.
     >RENUM SEQNUM WITH SOURCE
      Renum completed.
     >L A
        10     00010H
         SIMCAL
      6000     06000FINPUT    IP  F       72               DISK
      6001     06001FOUTPUT   O   F       72               DISK
     10000     10000IINPUT    AA  01   1  CA
     10001     10001I        OR  02   1  CS

(4)  >
     >RENUM SOURCE BY .1
      Error 153: Increment cannot be less than 1 for source.
     >RENUM SOURCE BY 10
      Renum completed.
     >

(5)  >L ALL
        10     00010H
         SIMCAL
      6000     00020FINPUT     IP   F       72                DISK
      6001     00030FOUTPUT    O   F       72                DISK
     10000     00040IINPUT     AA  01   1  CA
     10001     00050I         OR   02   1  CS
     >

(6)  >R SO W SO
      Error 159: Expecting SEQNUM after WITH.
     >R SO W SE
      Renum completed.
     >LIST A
        10     00010H
         SIMCAL
      6000     06000FINPUT     IP  F       72                DISK
      6001     06001FOUTPUT    O   F       72                DISK
     10000     10000IINPUT     AA  01   1  CA

(7)  10001     10001I         OR   02   1  CS
     >
```

RUN

RUN executes a prepared program.

Form

```
RU[N] progfile [,entrypoint] [;NOPRIV] [;LMAP] [;DEBUG]
                             [;MAXDATA=segsize]
                             [;PARM=num]
```

```
[;STACK=stacksize]
[;DL=dlsize]
[;NOCB]
     P
[;LIB=G]
     S
```

The parameters, preceded by a semicolon, can appear in any order. The RUN command is similar to the MPE :RUN command. Because it is similar, the syntax of RISE's RUN command is exactly the same as MPE's RUN command syntax. Therefore, delimiters between parameters are required.

Parameter Explanation

progfile — Actual designator of program file that contains prepared program.

entrypoint — Program entry point where execution is to begin. May be primary entry point of program or any secondary entry point in program's outer block. Default is primary entry point.

NOPRIV — Declaration that program segments will be placed in non-privileged mode capability. This parameter is intended for programs prepared with privileged mode capabilities. Normally, programs containing privileged instructions are executed in privileged mode only if the program was prepared with privileged mode. If NOPRIV is specified in the :RUN command, all program segments are placed in non-privileged mode. Library segments are not affected because their mode is determined independently. (Note that a program containing legally compiled privileged code, placed in a non-privileged mode, may abort when you attempt to execute it.)

LMAP — Request to produce a descriptive listing of the allocated (loaded) program on file whose formal designator is LOADLIST. If no :FILE command is found that references LOADLIST, listing is sent to $STDLIST. Default is no listing.

DEBUG — Request to issue a Debug call before the first executable instruction of the program. This parameter is ignored when a non-privileged user runs a program having privileged mode capability. This parameter is also ignored if the user does not have read and write access to the program file. Default is that DEBUG cal is not issued.

MAXDATA= — Maximum stack area (Z-DL) size permitted, in words.

segsize — This parameter is included if the size of DL-DB or Z-DB areas will be changed during the program execution. Default is that MPE assumes areas will not be changed.

PARM=num — Value that can be passed to program as a general parameter for control or other purposes. When program is executed, this value can be retrieved from address Q (initial) -4 where Q (initial) is Q address for outer block of program. Value can be octal number or signed or unsigned decimal number. Default is Q (initial) -4 address is filled with zeros.

stacksize — Size of initial local data area, Z-Q (initial), in stack. This value must exceed 511 words, and override stacksize estimated by MPE Segmenter. Default is estimated by Segmenter.

DL= dlsize — DL-DB area to be initially assigned to stack. This area is of interest mainly in programmatic applications. In all cases, the DL-DB area is rounded upward so that the distance from the beginning of the stack data segment to the DB address is a multiple of 128 words. Default is estimated by the Segmenter.

NOCB — Request that file system not use stack segment (PCBX) for its control blocks, even if sufficient space is available. This permits expansion of the stack (with the DLSIZE and ZSIZE intrinsics) to the maximum possible limit at a later time, but causes the file management system to operate more slowly for this program.

LIB=G Search segmented procedure libraries to satisfy external references during allocation in the following order: group library, account public library, system library. Default is S.

LIB=P Search segmented procedure library to satisfy external references during allocation in the following order: account public library, system library. Default is S.

LIB=S Search system library only to satisfy external references during allocation. This is the default.

Purpose

The RUN command allows you to execute a prepared program. It operates in the same way that the MPE command :RUN operates. Note that you can use the VERIFY RPG and VERIFY PREP commands to compile and prepare your work file into a program file to be executed with this RUN command. (See VERIFY.)

Execution mode: Line

Examples

The following are legal abbreviations which may be used with the RUN command and its parameters:

```
RU FCOPYPUB.SYS                 Run the FCOPY utility.

RU $OLDPASS;LIB= G; DEBUG       Run the prepared program $OLDPASS; search
                                segmented procedure libraries beginning
                                with the group library to satisfy
                                external references during allocation;
                                issue a DEBUG call before the first
                                executable instruction of the program.
```

In the example the RUN command is used to run two prepared programs. SIMCAL is to be used as the simple calculator it was designed to be, and it must be compiled, prepared, and run. Assume that SIMCAL was compiled into a USL file named $OLDPASS with RISE's VERIFY RPG command. Now it must be prepared using the Segmenter (or, alternatively, with the VERIFY PREP command). Initially, the command to run the Segmenter subsystem is given (1). The HP banner is then brought to the screen (2). The USL file is identified (3), and then prepared in a new program file (4). The user then leaves the Segmenter subsystem with the EXIT command and RISE's banner is displayed (5).

Next, the input and output file equations are given for SIMCAL. Note that the user must supply the colon (:) prompts (6). Following that, the RUN $OLDPASS executes SIMCAL (7).

Examples of SIMCAL's application follows: it is used to add (8), subtract (9), multiply (10), and divide (11). The colon entered when the calculations are finished returns the user to RISE (12).

```
(1)  >RUN SEGDVR.PUB.SYS

(2)  HP32050A.01.03 SEGMENTER (C) HEWLETT-PACKARD CO. 1979

(3)  -USL $OLDPASS

(4)  -PREP $NEWPASS

(5)  -EXIT

     HP32104A.05.00 RPG INTERACTIVE SYSTEM ENVIRONMENT   RISE
     (C) HEWLETT-PACKARD CO. 1981      WED, SEP 16, 1981, 10:29 AM

(6)  >:FILE INPUT=$STDIN
     >:FILE OUTPUT=$STDLIST
```

```
(7)  >RUN $OLDPASS

         SIMPLE             CALCULATOR

      ENTER DATA IN THE FORMAT:
      C NNNDD NNNDD

(8)   A 11100 11100

     ..... YOUR PROBLEM:  111.00 ADD   111.00 EQUALS      222.0000
      ENTER DATA IN THE FORMAT:
      C NNNDD NNNDD

(9)   S 22222 11111

..... YOUR PROBLEM:  222.22   SUB  111.11 EQUALS    111.1100

      ENTER DATA IN THE FORMAT:
      C NNNDD NNNDD

(10)  M 01000 01000

..... YOUR PROBLEM:   10.00  MULT   10.00 EQUALS    100.0000

      ENTER DATA IN THE FORMAT:
      C NNNDD NNNDD

(11)  D 10010 01000

..... YOUR PROBLEM:  100.10 DIV     10.00 EQUALS     10.0100

      ENTER DATA IN THE FORMAT:
      C NNNDD NNNDD

(12)  :

      HP32104A.05.00 RPG INTERACTIVE SYSTEM ENVIRONMENT   RISE
      (C) HEWLETT-PACKARD CO. 1981      WED, SEP 16, 1981, 10:29 AM

      >
```

SHOW

SHOW displays a page of your source file for direct editing on the screen.

Form

```
S[HOW] [lb] [C[OL]]
```

Parameter Explanation

lb First line of the page you wish to have displayed. (If omitted, the first page of the work file is displayed.)

COL Display column indicator rather than RPG Record Specification form (15 lines will be displayed with the COL parameter).

Purpose

The SHOW command displays a page (10 lines) for direct screen editing in Block Mode. If you do not specify a line number ("lb"), the first page of your work file will be displayed. "COL" is specified, a page of 15 lines is displayed with the column indicator. Otherwise, the page appears with the RPG Record Specification form appropriate to the line or lines being displayed. (An exception occurs when column 6 of a line contains an illegal RPG form entry. The screen display defaults to a column indicator in such an instance.)

In SHOW mode the terminal's cursor control keys and two of the editing keys--INSERT CHAR[acter] and DELETE CHAR[acter]--will prove particularly useful in your editing. (INSERT LINE and DELETE LINE keys are not activated.) Also, tabs have been set to correspond with the important Record Specification form columns. Using the TAB key rather than the space bar to position your cursor, you move quickly from column to column to the position where changes are necessary. You then type your changes in directly.

When you are satisfied with your changes, press the ENTER key to update the work file with your newly edited page.

Related Commands

Two sets of special function key labels can appear at the top of the screen with the SHOW command. They are the Master set and the Scroll set. To change sets, you press softkey F1, which is a toggle switch.

When the Master set is activated, the special function keys perform the following functions:

F1	Switch to Scroll set.
F2	Insert lines.
F3	Show page with or without edit sequence numbers.
F4	Move cursor to Command Window.
F5	Scroll forward to next page.
F6	Scroll backward one page.
F7	Enter Line Mode.
F8	Display main menu.

Softkey F3 will display the RPG source file on the screen with or without the sequence line numbers of the work file. If F3 is on, the sequence numbers will appear on the left hand side of the screen. RPG source records are 80 columns long. When RISE displays the source records on the screen without the sequence numbers, the individual records will fit on a single line. When you press F3, calling for a display of sequence numbers, columns 72-80 of each record are not shown. These columns are not deleted in such an instance, but the screen no longer contains the complete record since room is made to display the sequence numbers on the lefthand side.

Thus, you may display complete source records without the sequence numbers when you wish. If, however, you wish to determine where you are in the work file, and need sequence numbers to accomplish this, you will do so at the cost of truncating columns 72-80 from the screen display. They are not accessible at this time and cannot be altered.

Inserting Lines

When you wish to insert new lines between existing lines you may have to make room for them. If this is necessary you press F2 when the Master set of function keys is activated. This changes the special function key labels so that F1 through F4 are prepared to receive insertion commands. You then position the cursor at the line preceding the position where you wish to begin the insertion.

When the cursor is positioned correctly, you press softkeys F1, F2, F3, or F4, depending on how many lines you are adding (F1 adds 1 line; F4 adds 4 lines). An example of the correct procedure for inserting lines is included in the examples below.

NOTE If the terminal is unexpectedly locked out at this point, see the section on Recovery Procedures.

Scroll Set

While in SHOW mode, you can increase your scrolling capabilities by switching to Scroll set. Again, this is accomplished by pressing F1.

The softkeys have the following functions when Scroll set is activated:

F1 Switch to Master set.

F2 Display first page.

F3 Display last page.

F4 Move cursor to Command Window.

F5 Scroll forwards a page.

F6 Scroll backwards a page.

F7 Scroll forwards 1/2 page.

F8 Scroll backwards 1/2 page.

Execution mode: Block.

Examples

The following are legal abbreviations which may be used with the SHOW command and its parameters:

S F C Enter Show mode displaying the first line of the source file with the column indicator.

S 10 Enter Show mode and display line 10 with the appropriate Record Specification form.

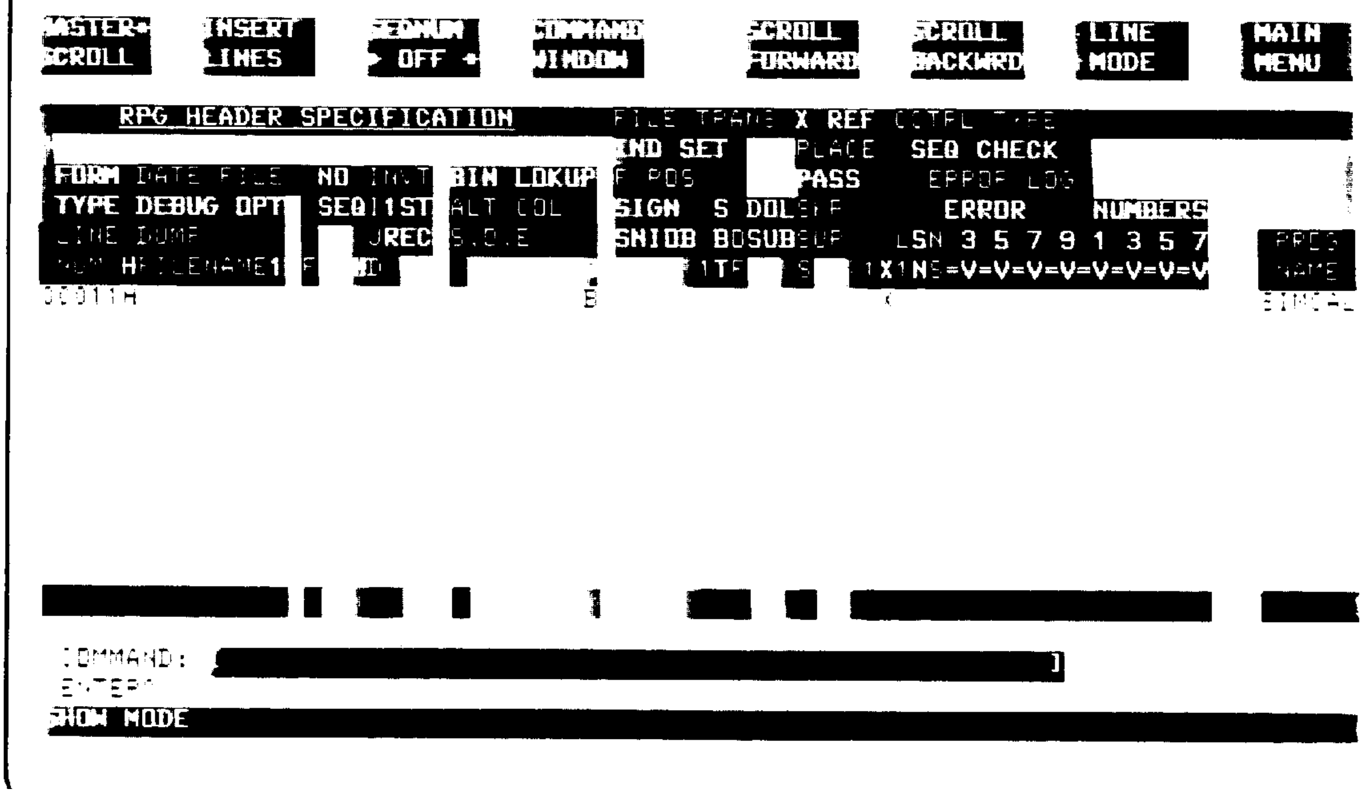
SHO Enter Show mode.

In the first example, the SHOW FIRST command brings the Header Specification form to the screen along with the first line of the source file (1). In the second display, the cursor has been moved into position, and new data is added to the first line. Note that RISE is now in Block mode, and the ENTER key must be pressed to make changes to the first line permanent (2).

```
> SHOW FIRST
```

Bad graphic

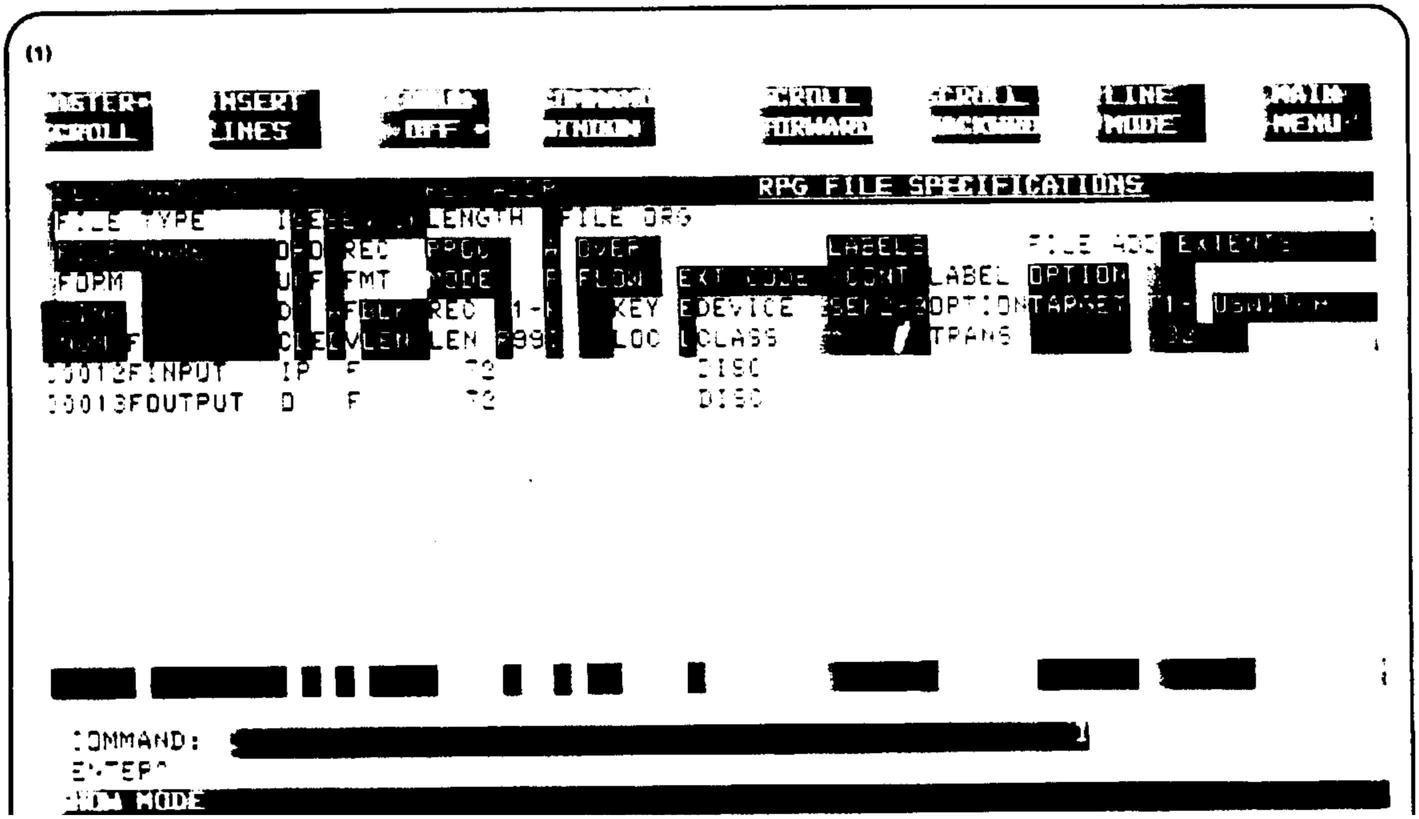
(2)



```
    RPG HEADER SPECIFICATION        FILE TRANS X REF CCTRL TYPE
                                    IND SET      PLACE   SEQ CHECK
FORM DATE FILE   NO INVT  BIN LOKUP F POS        PASS     ERROR LOG
TYPE DEBUG OPT   SEQ 1 1ST ALT COL  SIGN  S DOLSIF        ERROR     NUMBERS
LINE DUMP            JREC S.O.E     SNIOB BOSUBSUP     LSN 3 5 7 9 1 3 5 7    PROG
NUM HFILENAME1  F  ND                  1TF   S     1X1NS=V=V=V=V=V=V=V=V      NAME
00011H                       B                      (                          SIMCAL

COMMAND:
ENTER:
SHOW MODE
```

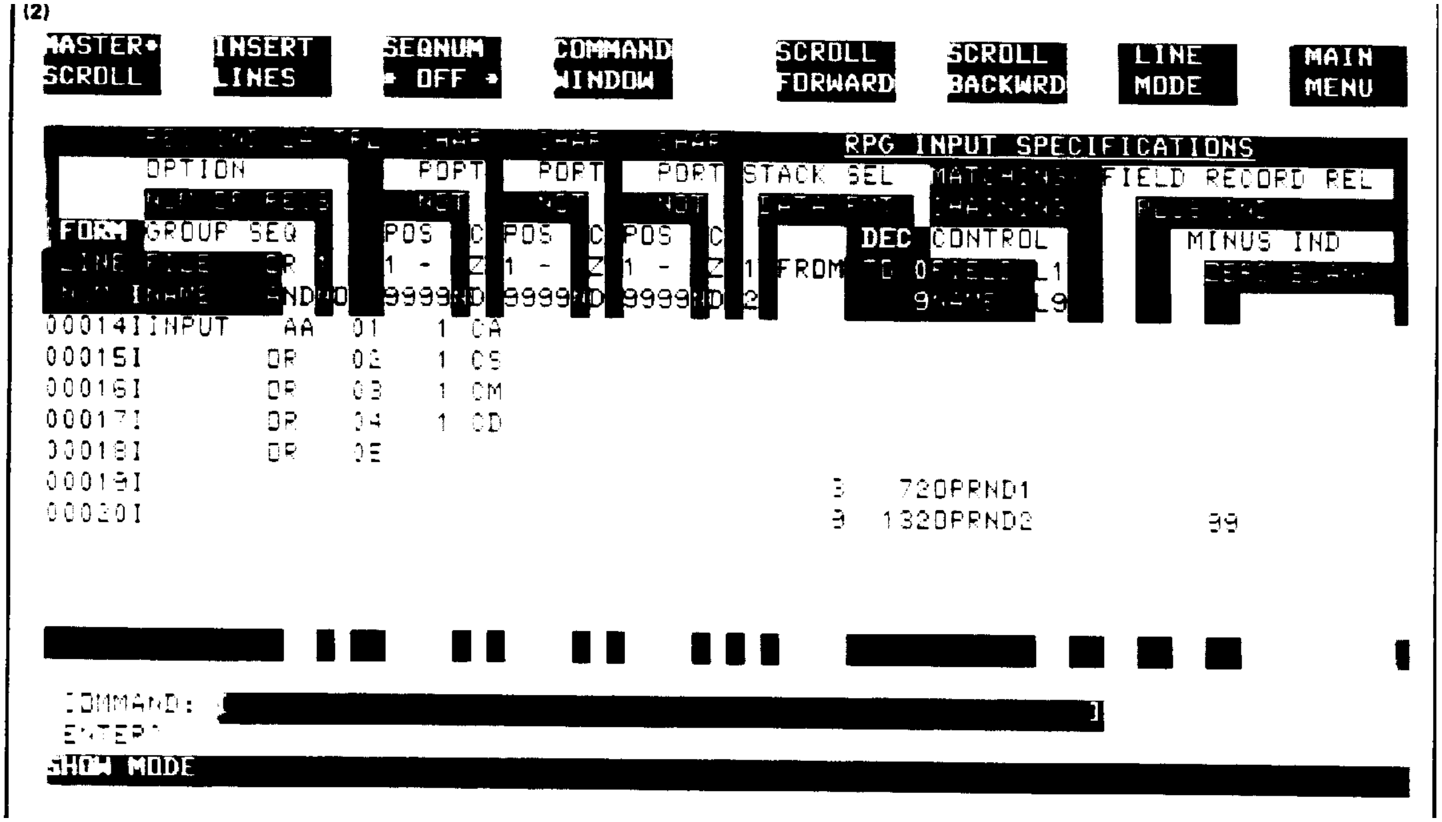
In the next examples, the special function key labeled SCROLL FORWARD is used while the Header Specification form is displayed. The first time the scrolling occurs, the RPG File Specification form appears on the screen with the associated source file lines (1). The next form to be brought to the screen by the SCROLL FORWARD softkey is the RPG Input Specification form. Again, the associated lines are also displayed (2).

(1)



```
RPG FILE SPECIFICATIONS
FILE TYPE          LENGTH   FILE ORG
                   REC  PROC    OVER                 LABELS          FILE ADD EXTENTS
FORM               FMT  MODE    FLOW    EXT CODE     CONT LABEL OPTION
                   REC  1-          KEY EDEVICE          OPTION TARGET T1-
                   LEN  LEN 999     LOC  CLASS               TRANS
00012FINPUT    IP   F       72          DISC
00013FOUTPUT   O    F       72          DISC

COMMAND:
ENTER
```

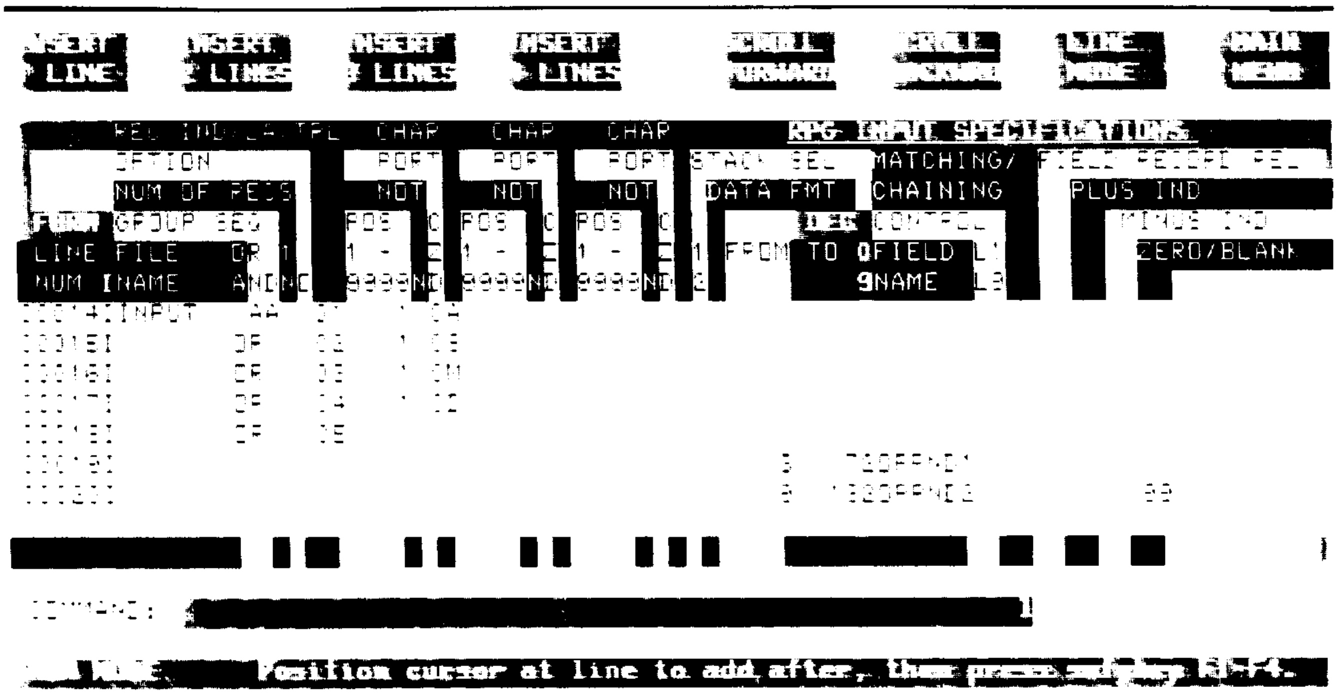
(2)



```
                                                        RPG INPUT SPECIFICATIONS
      OPTION              PORT      PORT      PORT  STACK SEL               FIELD RECORD REL
FORM  GROUP SEQ       POS   C POS   C POS   C                DEC CONTROL        MINUS IND
                      1 -   Z 1 -   Z 1 -   Z 1 FROM                L1
            ANDNO     9999ND  9999ND  9999ND  2                     L9
00014IINPUT    AA  01    1 CA
00015I         OR  02    1 CS
00016I         OR  03    1 CM
00017I         OR  04    1 CD
00018I         OR  0E
00019I                                                  3   720PRND1
00020I                                                  9  1320PRND2           99

COMMAND:
ENTER
```

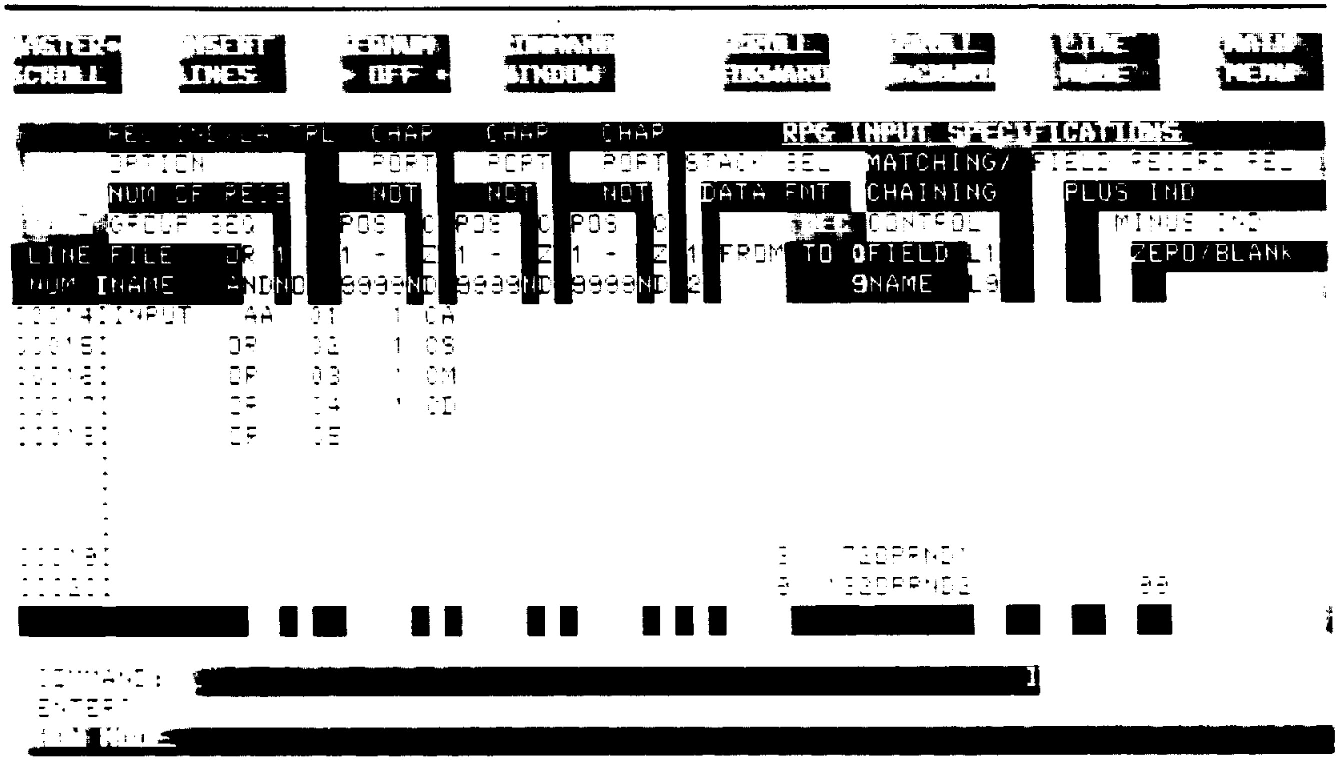
In the next example, the form shown with the message in the bottom line communication window, and the changed labels for the softkeys are produced when the INSERT LINES softkey (F2) is pressed. Note that the SHOW MODE window contains precise instructions for inserting from one to four lines (1). The second display is produced by pressing softkey #3 (INSERT 3 LINES) after the cursor has been positioned at line 18. Note that the column 6 entries, which designate the Record Specification form appropriate to the lines being inserted, are supplied automatically by RISE (2). The user is then ready to position the cursor and insert new data.

1)



```
     REC IND/LA TPL  CHAR   CHAR   CHAR    RPG INPUT SPECIFICATIONS
     OPTION          PORT   PORT   PORT STACK SEL MATCHING/ FIELD RECORD REL
     NUM OF RECS     NOT    NOT    NOT   DATA FMT CHAINING   PLUS IND
FORM GROUP SEQ       POS  C POS  C POS  C    DEC  CONTROL      MINUS IND
LINE FILE     OR 1   1 -  C 1 -  C 1 -  C 1 FROM TO 0FIELD L1  ZERO/BLANK
NUM  INAME    ANDNC  9999ND 9999ND 9999ND 2         9NAME  L9
00014IINPUT   AA     01       1 CH
00015I        OR     02       1 CS
00016I        OR     03       1 CM
00017I        OR     04       1 CD
00018I        OR     05
00019I                                    3    7IDPRNO1
00020I                                    8   10IDPRNO2        99

COMMAND:

NOTE: Position cursor at line to add after, then press softkey F1-F4.
```

2)



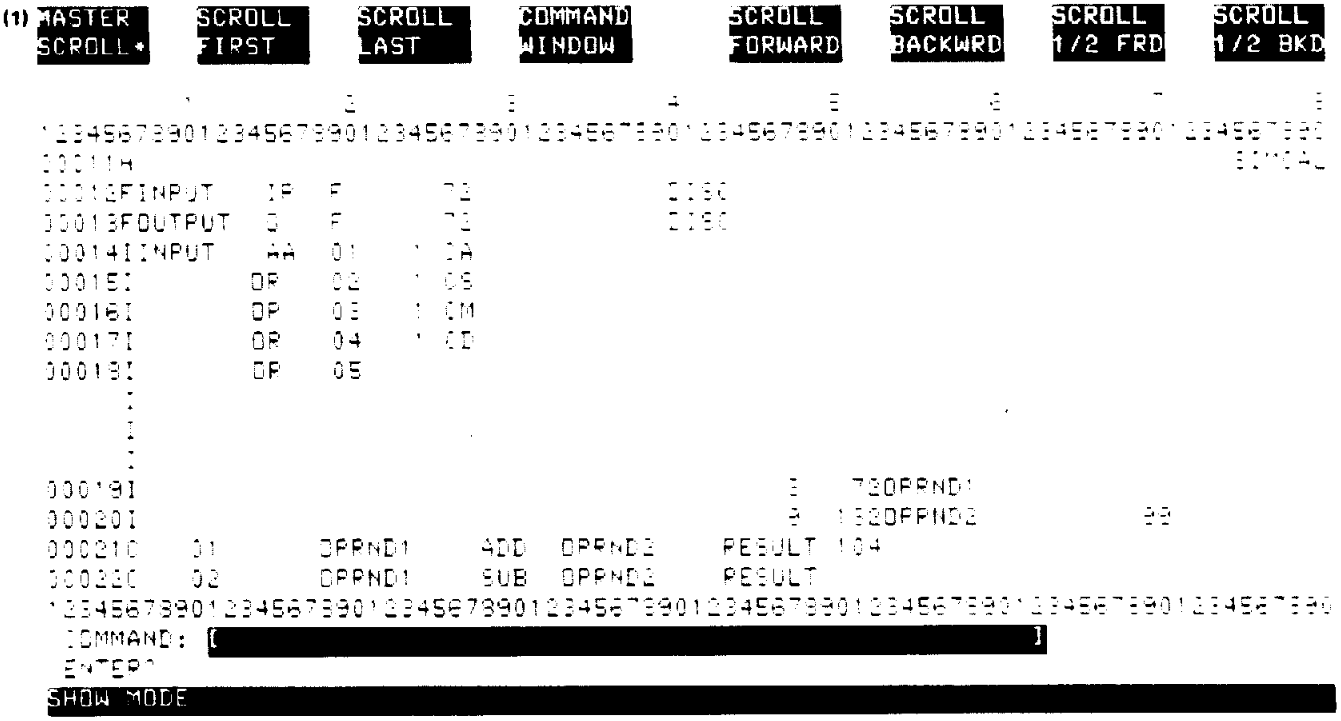
```
     REC IND/LA TPL  CHAR   CHAR   CHAR    RPG INPUT SPECIFICATIONS
     OPTION          PORT   PORT   PORT STACK SEL MATCHING/ FIELD RECORD REL
     NUM OF RECS     NOT    NOT    NOT   DATA FMT CHAINING   PLUS IND
FORM GROUP SEQ       POS  C POS  C POS  C    DEC  CONTROL      MINUS IND
LINE FILE     OR 1   1 -  C 1 -  C 1 -  C 1 FROM TO 0FIELD L1  ZERO/BLANK
NUM  INAME    ANDNC  9999ND 9999ND 9999ND 2         9NAME  L9
00014IINPUT   AA     01       1 CH
00015I        OR     02       1 CS
00016I        OR     03       1 CM
00017I        OR     04       1 CD
00018I        OR     05
     .
     .
     .
     .
     .
     .
     .
00019I                                    3    7IDPRNO1
00020I                                    8   10IDPRNO2        99

COMMAND:
ENTER
```

In the next example, the screen is switched to Scroll Set following the insertion by pressing special function key #1 (MASTER SCROLL). Note that the asterisk in the softkey label for F1 is now next to

"SCROLL". This indicates that the Scroll Set of keys are now active (1). Special function key #3 (SCROLL LAST) produces the next screen display, which contains the final lines of the source file associated with the RPG Output Specifications form (2).

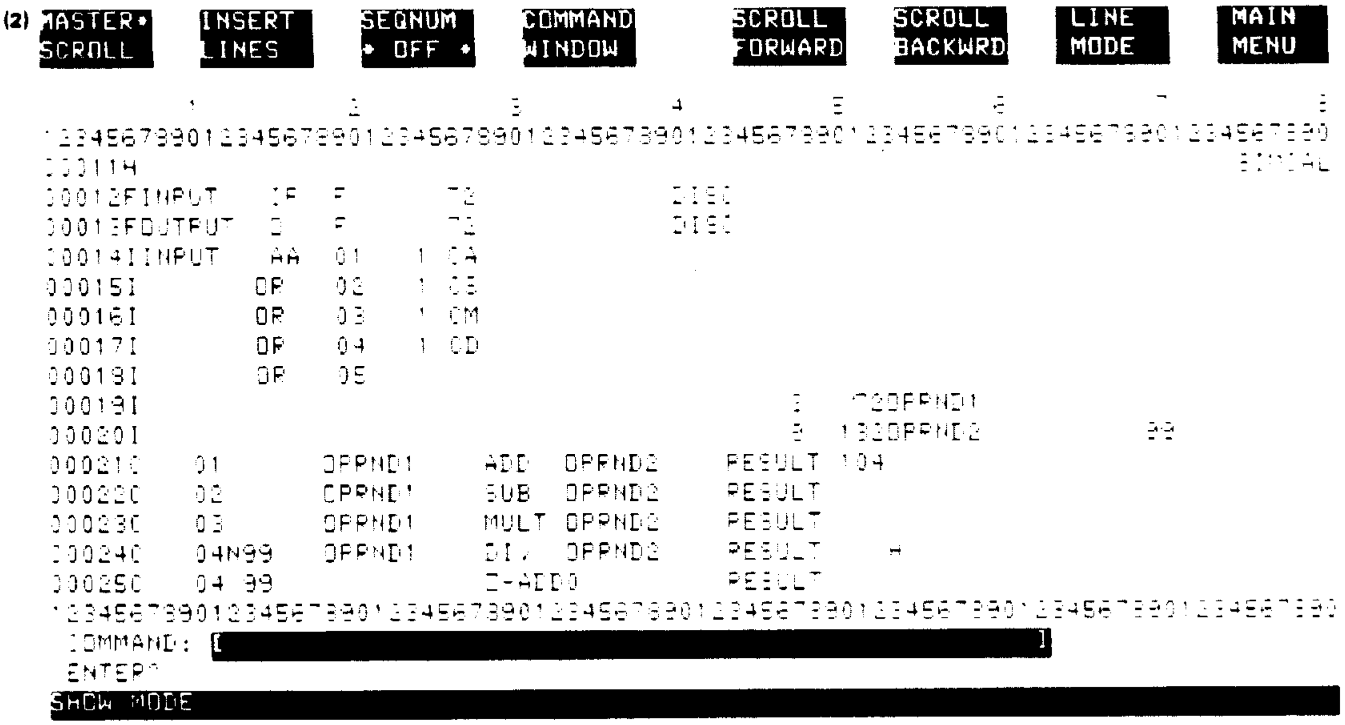
Bad graphic

In the next example, the SHOW FIRST COL command brings the column indicator to the screen. In the first display, Scroll set is activated as the softkey labels indicate (1). Pressing special function key #1 (MASTER SCROLL) brings the next display to the screen. Note that the asterisk is now next to the word "MASTER." RISE is again in Show mode, Master Set, with the column indicator shown (2). (The spaces created between lines 18 and 19 by the insertion example have been deleted by erasing the "I's" in column 6 and then pressing the ENTER key to make the changes permanent in the work file. If they were not, the spaces would be retained on the screen.)



(1)

```
MASTER     SCROLL     SCROLL     COMMAND        SCROLL     SCROLL     SCROLL     SCROLL
SCROLL*    FIRST      LAST       WINDOW         FORWARD    BACKWRD    1/2 FRD    1/2 BKD

         1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890
00011H                                                                    SIGNAL
00012FINPUT   IP   F        72             DISC
00013FOUTPUT  O    F        72             DISC
00014IINPUT    AA   01    1 CA
00015I        OR    02    1 CS
00016I        OR    03    1 CM
00017I        OR    04    1 CD
00018I        OR    05
     .
     .
     I
     .
     .
00019I                                                3    72OPRND1
00020I                                                9  132OPRND2         99
00021C    01        OPRND1     ADD  OPRND2     RESULT 104
00022C    02        OPRND1     SUB  OPRND2     RESULT
12345678901234567890123456789012345678901234567890123456789012345678901234567890
 COMMAND: [                                          ]
 ENTER?
SHOW MODE
```

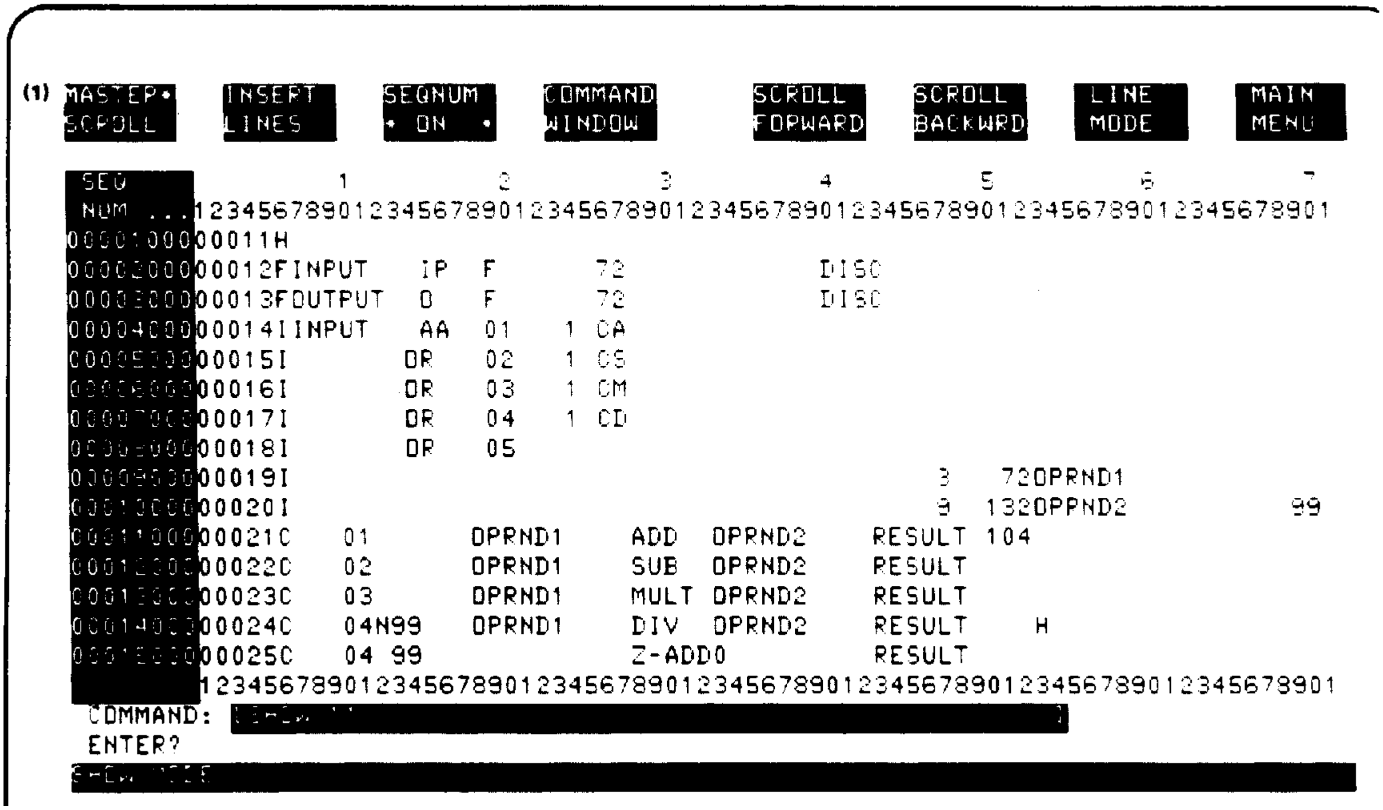


(2)

```
MASTER*    INSERT     SEQNUM     COMMAND        SCROLL     SCROLL     LINE       MAIN
SCROLL     LINES      * OFF *    WINDOW         FORWARD    BACKWRD    MODE       MENU

         1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890
00011H                                                                    SIGNAL
00012FINPUT   IP   F        72             DISC
00013FOUTPUT  O    F        72             DISC
00014IINPUT    AA   01    1 CA
00015I        OR    02    1 CS
00016I        OR    03    1 CM
00017I        OR    04    1 CD
00018I        OR    05
00019I                                                3    72OPRND1
00020I                                                9  132OPRND2         99
00021C    01        OPRND1     ADD  OPRND2     RESULT 104
00022C    02        OPRND1     SUB  OPRND2     RESULT
00023C    03        OPRND1     MULT OPRND2     RESULT
00024C    04N99     OPRND1     DIV  OPRND2     RESULT    H
00025C    04 99                Z-ADD0          RESULT
12345678901234567890123456789012345678901234567890123456789012345678901234567890
 COMMAND: [                                          ]
 ENTER?
SHOW MODE
```

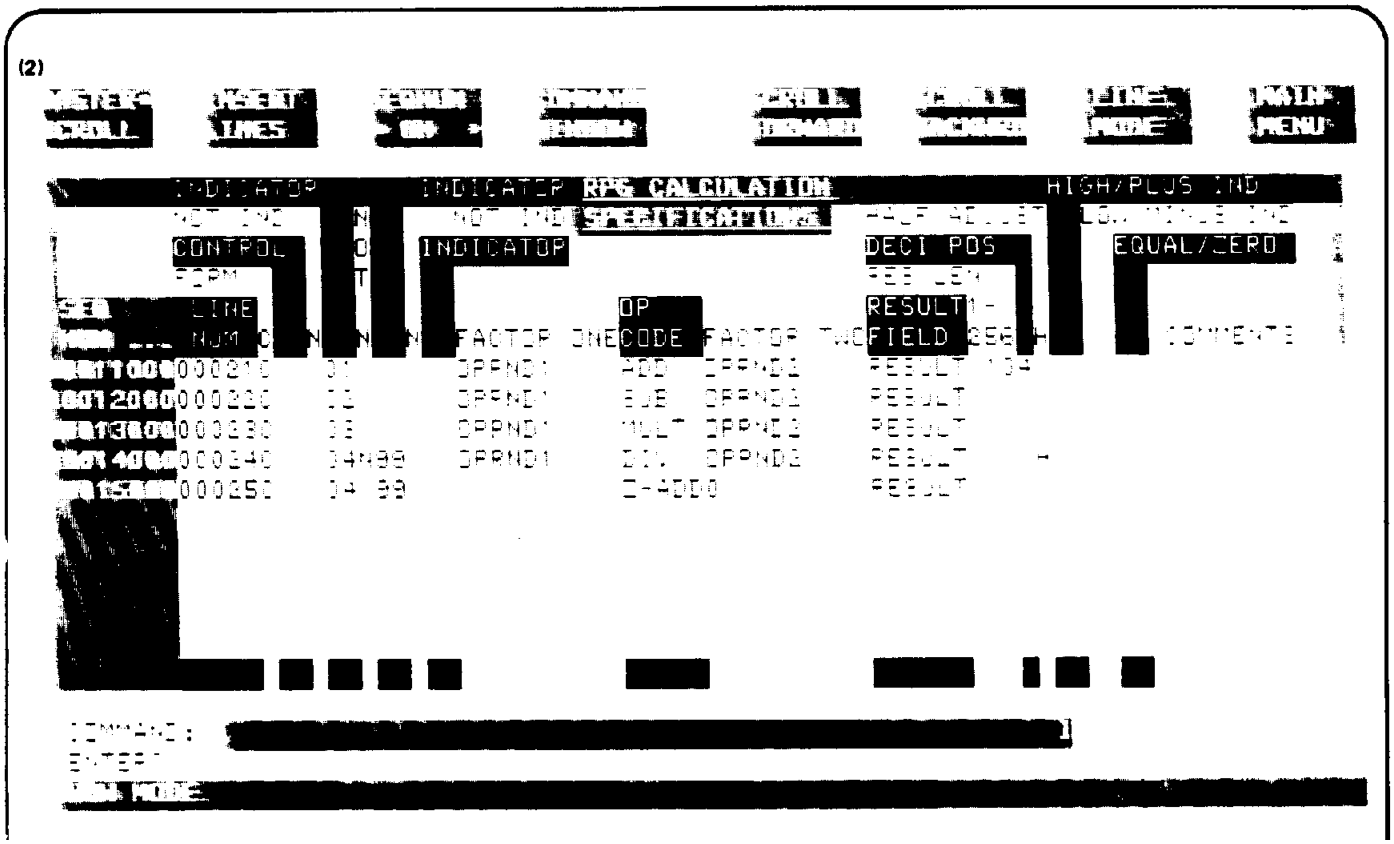
To produce the first display in the final example, the user has entered SHOW FIRST COL command or some valid abbreviation. Also, special function key #3 has been pressed while in Master Set (this key toggles sequence numbers on and off). A fifteen line "page" is brought to the screen and with it the sequence line numbers which RISE keeps for use in referencing records. Note the command in the command window (1). The next display follows the execution of the command, SHOW 11, after the user hits the ENTER key. The column indicator form is cleared and the RPG Calculation Specifications form, which is appropriate to line 11, is brought to the screen. Sequence line numbers continue to be displayed and will be until special function key #3 is pressed (2).

(1)



```
 SEQ             1         2         3         4         5         6         7
 NUM ...1234567890123456789012345678901234567890123456789012345678901234567890
0000100000011H
0000200000012FINPUT   IP  F      72            DISC
0000300000013FOUTPUT  O   F      72            DISC
0000400000014IINPUT   AA  01   1 CA
0000500000015I       OR   02   1 CS
0000600000016I       OR   03   1 CM
0000700000017I       OR   04   1 CD
0000800000018I       OR   05
0000900000019I                                        3  72OPRND1
0001000000020I                                        9 132OPRND2           99
0001100000021C   01      OPRND1    ADD  OPRND2    RESULT 104
0001200000022C   02      OPRND1    SUB  OPRND2    RESULT
0001300000023C   03      OPRND1    MULT OPRND2    RESULT
0001400000024C   04N99   OPRND1    DIV  OPRND2    RESULT    H
0001500000025C   04 99             Z-ADD0         RESULT
        1234567890123456789012345678901234567890123456789012345678901234567890
 COMMAND:
 ENTER?
SHOW MODE
```

(2)



```
          INDICATOR          INDICATOR RPG CALCULATION            HIGH/PLUS IND
          NOT IND      N      NOT IND  SPECIFICATIONS   HALF ADJUST LOW/MINUS IND
          CONTROL      O    INDICATOR                   DECI POS       EQUAL/ZERO
          FORM         T                                RES LEN
SEQ       LINE                            OP            RESULT1-
NUM ...   NUM C N    N    N FACTOR ONE CODE FACTOR TWO FIELD 356 H       COMMENTS
00011000000210   01        OPRND1     ADD  OPRND2      RESULT 104
00012000000220   02        OPRND1     SUB  OPRND2      RESULT
00013000000230   03        OPRND1     MULT OPRND2      RESULT
00014000000240   04N99     OPRND1     DIV  OPRND2      RESULT    H
00015000000250   04 99                Z-ADD0           RESULT

 COMMAND:
 ENTER?
[illegible] MODE
```

TEXT

TEXT copies a permanent file into a work file for editing.

Form

```
         T[EXT] filename [N[OW]]
```

Parameter Explanation

filename Name of the permanent file you are copying to the work file.

N[OW] Your work file contains the contents of one file but you may wish to TEXT another without saving the contents of that work file. The contents of the file you wish to TEXT will not be copied unless you KEEP the current work file to a permanent file. You can override the need to KEEP the file by using the NOW parameter. In doing so you discard your changes. If you are directly editing a file (using the FILE command) and you give the TEXT command, the file will be saved (closed) automatically, whether you specify "NOW" or not.

If you choose not to use the "NOW" parameter, RISE displays a safety message asking you if it is all right to clear the current work file. You may respond with "Y[ES]" to clear the current file and text in another. If you respond with "N[O]", the TEXT command is cancelled so that you may perform a KEEP first if you wish.

Purpose

The TEXT command copies the contents of a permanent file into a work file for additions and changes. You make additions and changes to a work file only, which you save after editing with the KEEP command.

Related Commands

See the discussion of the KEEP command. If the file is unnumbered, RISE automatically appends the sequence numbers in columns 81-88. If the text file is already numbered, RISE uses the existing sequence numbers. Text files are compatible with EDIT/3000 files.

```
 *  Execution mode:  Line or Block.

 *  Record pointer:  At first line in work file.
```

Examples

The following are legal abbreviations which may be used with the TEXT command and its parameters:

```
T FILE1 N         Copy the contents of a permanent file named FILE1 into
                  the work file immediately, without displaying safety
                  prompts or saving changes made to the work file.

TEX FILE1         Copy the contents of the permanent file named FILE1 into
                  the work file, displaying safety prompts if the current
                  work file has been edited and has not be saved with the
                  KEEP command.
```

In the first example, SIMCAL is copied into the work file using the TEXT command, and an execution message is returned (1). The first five lines of the file are listed, and then lines 1 through 3 are deleted (2). An attempt to TEXT SIMCAL with these changes receives a safety prompt (3). When it is answered negatively, the command is cancelled (4). Next, the KEEP command is used with the filename BACKUP. Thus, BACKUP without the three lines, is now stored as a permanent file (5).

```
     (1)  >TEXT SIMCAL
           Text completed.
          >L 1/5
              1     00011H
              SIMCAL
              2     00012FINPUT    IP  F       72               DISK
              3     00013FOUTPUT   O   F       72               DISK
              4     00014IINPUT    AA  01   1  CA
              5     00015I        OR   02   1  CS

     (2)  >D 1/3
              1     00011H
              SIMCAL
              2     00012FINPUT    IP  F       72               DISK
              3     00013FOUTPUT   O   F       72               DISK
           Delete completed.

     (3)  >T SIMCAL
           KEEP not done, CLEAR current work file?NO
           Text cancelled.

     (4)  >KEEP BACKUP
           Keep completed.

     (5)   File name is BACKUP
```

In the second example, the CHANGE command is used to replace one four-character string with another. The results of the command, and an execution message, are then displayed (1). The TEXT command is used with the NOW parameter and is executed (2). Note that no safety prompts were issued even though changes were previously made. When LIST 1/8 is given, the first three lines are missing from the listed lines because they were deleted in the first example (3). Next, line 5 is modified (4). The TEXT command used with SIMCAL evokes the safety prompts because of the changes made to the file. When the prompt is answered in the affirmative, the command is executed (5).

```
     >CHANGE ;DISK; TO ;TAPE; IN ALL
         2     00012FINPUT    IP  F       72   TAPE
         3     00013FOUTPUT   O   F       72   TAPE
(1)   Changes completed.
     >TEXT BACKUP NOW

(2)   Text completed.
     >

(3)  >L1/8
         4     00014IINPUT    AA  01   1  CA
         5     00015I        OR   02   1  CS
         6     00016I        OR   03   1  CM
         7     00017I        OR   04   1  CD
         8     00018I        OR   05

(4)  >MS
         5     00015I         OR   02    1 CS
              :              RAND
         5     00015I        AND   02    1 CS
              :

(5)  >TEXT SIMCAL
      KEEP not done, CLEAR current work file?YES
      Text completed.
     >
```

UNDEL

UNDELete restores lines that were erroneously deleted.

Form

```
U[NDEL]
```

Purpose

The UNDELete command restores the lines which were removed by the last DELETE command. It does not restore characters which were deleted while in the MODIFY mode.

If you DELETE lines, then ADD new ones with the same numbers as those deleted, the deletions will not be restored by the UNDEL command. It only restores lines when the numbers of the lines to be restored do not already exist in the work file. For example, if you DELETE lines 1/5, ADD 1/3, then try to recover with the UNDEL command, RISE will not be able to restore lines 1/3 because the lines exist due to your new additions. In this case, no lines will be restored unless you move 1/3 to another location, freeing those locations for full restoration.

Related Commands

In SHOW Mode, changes are not permanent until the ENTER key is depressed, at which time RISE changes the internal page buffer and updates the work file with it. If, previous to pressing the ENTER key, you made changes which you wish to cancel, you do not use the UNDEL command. You scroll forward or backward to a different page, then scroll back to the page on which the undesired changes were made. RISE will restore that page to its original state from the internal buffer which never changed.

* Execution mode: Line or Block.

* Record pointer: At last line undeleted.

* Control Y: Stops listing the lines that are being undeleted.

Examples

The following are legal abbreviations which may be used with the UNDELete command:

U Undelete lines removed by the last delete command.

UND

In the first example, lines 1 through 5 of SIMCAL are listed and then deleted. An execution message appears at the terminal when the deletion is completed (1). The UNDEL command is then used, and the lines previously deleted are restored as they were (2). Lines 5 through 10 are next deleted (3).

```
          >L1/5
              1       00011H
              SIMCAL
              2       00012FINPUT    IP  F       72               DISK
              3       00013FOUTPUT   O   F       72               DISK
              4       00014IINPUT    AA  01   1  CA
              5       00015I        OR   02   1  CS
          >D1/5
              1       00011H
              SIMCAL
              2       00012FINPUT    IP  F       72               DISK
              3       00013FOUTPUT   O   F       72               DISK
              4       00014IINPUT    AA  01   1  CA
              5       00015I        OR   02   1  CS
     (1)   Delete completed.
          >
          >UNDEL
              1       00011H
              SIMCAL
              2       00012FINPUT    IP  F       72               DISK
              3       00013FOUTPUT   0   F       72               DISK
```

```
         4       00014IINPUT    AA  01   1  CA
         5       00015I        OR   02   1  CS

(2)   Undel completed.
     >
     >D 5/10
         5       00015I        OR   02   1  CS
         6       00016I        OR   03   1  CM
         7       00017I        OR   04   1  CD
         8       00018I        OR   05
         9       00019I                     3  72OPRND1
        10       00020I                     9 132OPRND2     99

(3)    Delete completed.
```

In the next example, a file named RPGDOC--used previously in the JOIN example--is JOINed to the work file beginning at line 5 (1). The UNDEL command is then tried and rejected by RISE. This occurs because line 5, which was deleted in the first example, has been used again through the JOIN command. A message appears explaining why the UNDELete command cannot be executed (2). The lines containing RPGDOC are then MOVEd to a new position, beginning at line 10.5 (3). When this is done, line 5 no longer exists in the file, Now the UNDELete command, which was previously rejected, is executed by RISE (4).

```
     >JOIN RPGDOC TO 5
         5          H*
         5.1        H*  PROGRAM NAME:
         5.2        H*
         5.3        H*  PROGRAMMER  :
         5.4        H*  DATE WRITTEN:
         5.5        H*  PURPOSE     :
         5.6        H*
(1)   Join completed.

(2)  >UNDEL
         5     00015I      OR     02  1  CS
      Can't Un-delete a line, the line number exists already.
     >

(3)  >MOVE 5/5.6 TO 10.5
        10.5        H*
        10.51       H*  PROGRAM NAME:
        10.52       H*
        10.53       H*  PROGRAMMER  :
        10.54       H*  DATE WRITTEN:
        10.55       H*  PURPOSE     :
        10.56       H*
      Move completed.

(4)  >U
         5     00015I       OR   02   1  CS
         6     00016I       OR   03   1  CM
         7     00017I       OR   04   1  CD
         8     00018I       OR   05
         9     00019I                3 72OPRND1
        10     00020I                9 1320PRND2      99

       Undel completed.
     >
```

VERIFY

VERIFY compiles a work file, lists a compilation, or prepares a USL file into a program file.

Form

```
              R[PG]
```

```
V[ERIFY] L[IST]
         P[REP] [progfile] [;ZERODB] [;PMAP]
                [;MAXDATA=segsize] [;STACK=stacksize]
                [;DL=dlsize] [;CAP=caplist] [;RL=filename]
                [;PATCH=patchsize]
```

Parameter Explanation

R[PG] Calls the RPG compiler to compile the work file into the USL file named "$OLDPASS", a system-defined file name. Afterwards, it causes the newly compiled listing to be displayed on the screen. Using special function keys, you can scroll through the file in split screen. The USL file in $OLDPASS is not created if any serious compilation errors occur.

L[IST] Displays the last compilation listing completed as a result of the V[ERIFY] R[PG] command.

P[REP] Calls the Segmenter to prepare a USL file in $OLDPASS to the program file named "progfile". If "progfile" is omitted, the program file will default to $OLDPASS. (During the preparation of the USL file, the Segmenter uses $NEWPASS as the program file name. Afterwards, $NEWPASS is renamed $OLDPASS.)

progfile The name of the file onto which the prepared segments are to be written. If the file named does not exist, the Segmenter will build a job temporary file for you. (Note: code segments in a program file cannot lie across disk extent boundaries. Thus, all segments in such files must be constructed within one extent. See the MPE Intrinsics Reference Manual for a discussion of disk extents.)

ZERODB An indication that the initially-defined DL-DB area, and uninitialized portions of DB-Q (initial) area will be initialized to zero. If this parameter is omitted, these areas are not affected.

PMAP An indication that a listing describing the prepared program will be produced with the formal filename of "SEGLIST". If there is no file equation for "SEGLIST", the listing is printed on your terminal. If the parameter is omitted, no listing is produced.

MAXDATA= Maximum stack area (Z-DL) size permitted, in words.

segsize This parameter is included if you expect to change the size of the DL-DB or DB-Z areas during process execution. If omitted, MPE assumes that these areas will not be changed.

STACK= The size of the user's initial local data area, Q

stacksize (initial) to Z, ior the stack, in words. This overrides the stacksize estimated by the Segmeorter, which applies if the stacksize parameter is omitted. (The default is a function of estimated stack requirements for each program unit in the program.) Since it is difficult for the system to predict the behavior of the stack at runtime, you may want to override the default by supplying your own estimate with stacksize.

DL=dlsize The DL-DB area to be initially assigned to the stack. If the "dlsize" parameter is omitted, a value of zero is used.

CAP=caplist The capability-class attributes associated with the caplist user's program; specified as two-character mnemonics. If more than one mnemonic is specified, each must be separated from its neighbor by a comma.

The mnemonics are:

- IA = Interactive access.
- BA = Local batch access.

* PH = Process handling.

* DS = Data segment management.

* MR = Multiple resource management.

* PM = Privileged-mode operation.

Users who issue the prepare command can only specify capabilities that they themselves possess (through assignment by the Account Manager). If the user does not specify any capabilities, only IA and BA (if the user possesses them) will be assigned to this program.

RL=filename The name of the Relocatable Library (RL) to be searched to satisfy external references during preparation. This can be any permanent file of type RL. It need not belong to the log-on group, nor does it have a reserved, local name. This file yields a single segment that is incorporated into the segments of the program file prepared. If "filename" is omitted, no library will be searched.

PATCH=patchsize The number of extra words added to each code segment before the Segment Transfer Table to allow room for code patches to a segment.

For the stacksize, dlsize, and maxdata parameters, a value of -1 denotes the default (equivalent to omitting the parameter).

NOTE Note that the optional parameters associated with the P[REP] command can be used in any order. When the Segmenter is executing, its HP banner and all its responses will appear on your terminal. Moreover, because RISE's PREP option is exactly the same as the Segmenter's PREPARE command, the syntax is the same, and delimiters are necessary between parameters.

Purpose

The VERIFY command has three functions. It calls the RPG compiler to compile the work file, and when the compilation is finished, displays the newly compiled file listing. Also, it lists the last compilation listing produced by the VERIFY command and its RPG parameter. Finally, it calls the Segmenter to prepare a USL file in $OLDPASS to a program file. Therefore, you may compile, prep, and execute your RPG source program in the work file by using the VERIFY RPG, VERIFY PREP, and RUN commands. You may also use the :MPEcommand to set up any file equations necessary to execute your program correctly. For instance:

```
:FILE OUTENTITY =$STDLIST
```

Related Commands

The special function keys are redefined by the VERIFY RPG and LIST commands to facilitate management of the compilation listing. The keys function as follows:

F1 Switch window to split screen--Top or Bottom--or Full Window.

F2 Display first page of compilation listing.

F3 Display last page.

F4 Find error in compilation listing.

F5 Scroll forwards one page.

F6 Scroll backwards one page.

F7 Scroll forwards 1/2 page.

F8 Scroll backwards 1/2 page.

Softkey F1 allows you to divide the screen up into two fixed portions--a top and bottom half--or join them in a full page display. Pressing F1 will switch the screen display in the following order: Top Window, Bottom Window, and back to Full Window. The F1 label indicates what will be displayed next on the screen if you press F1. In other words, when Top Window is activated, the label will read Bottom Window. If you press F1, RISE will respond by activating the Bottom Window.

Futhermore, a dividing line separating the two halves of the screen indicates the activated half. Up arrows (^) point toward the Top Window; a line of v's indicates the Bottom Window.

Use of Split Screen

The split screen allows you to view, simultaneously, two parts of the compilation listing at one time. For example, one window can show error messages while the other shows the corresponding source line containing the error.

Special function key F4, the "Find Error" key, is extremely useful when you use it with Top Window mode. After you press F4, RISE will display the next RPG source line with errors or warnings in the Top Window while automatically displaying the corresponding error message at the bottom.

Softkey F4 finds and displays the next source line containing an RPG error or warning. The search for the error begins after the last line displayed on the current window. F4 only finds errors for which the RPG compiler emits an error number, and only on the source line containing that error.

PRINT: the PRINT command with the RPGLIST parameter will print the compilation listing offline.

```
                               RPG
     Execution mode: VERIFY          Block mode
                               LIST

                     VERIFY    PREP  Line mode
```

Examples

The following are legal abbreviations and can be used with the VERIFY command and its parameters:

V R The RPG compiler will compile the edit file.

V L The last compilation-listing produced by the V[ERIFY] R[PG] command will be displayed on the screen.

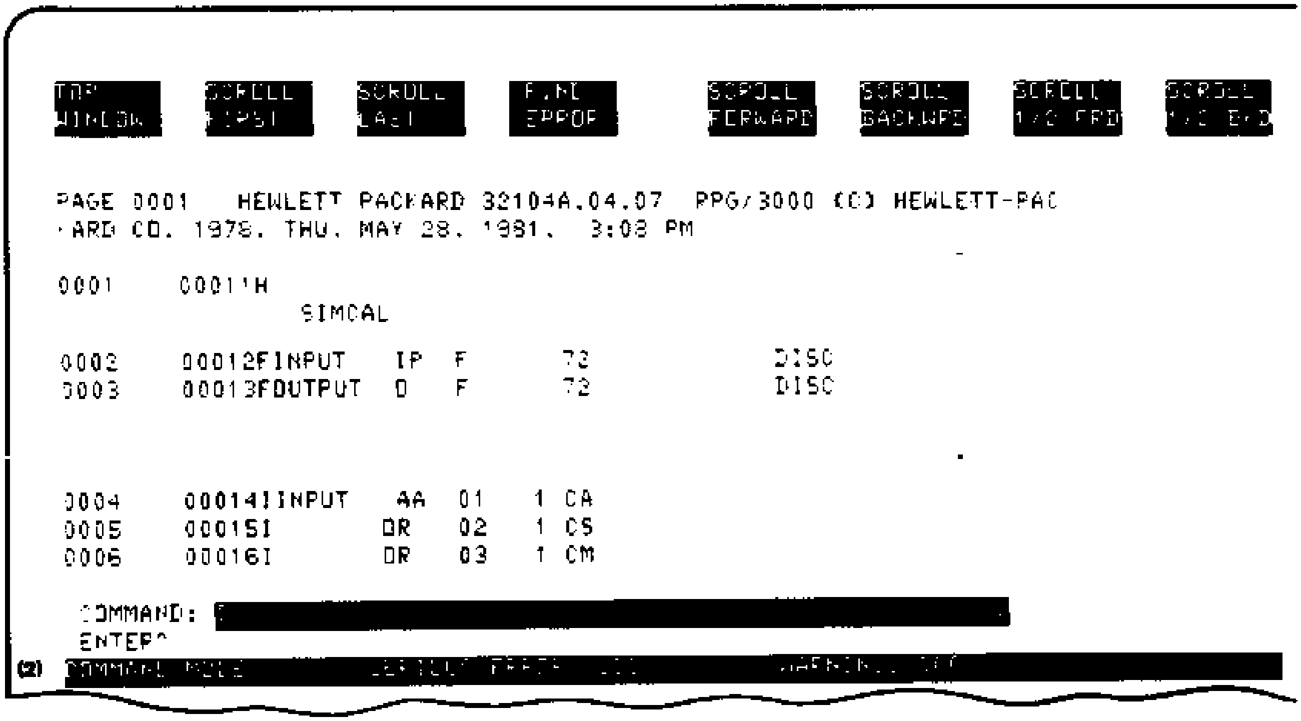
V PREP The segmenter will prepare a USL file in $OLDPASS into a program file.

In the first example, five lines of SIMCAL are listed on the screen. The work file containing SIMCAL is then compiled when the VERIFY RPG command is given, and the results are displayed (1). Note that the Command Mode window contains a message concerning serious errors and warnings after the VERIFY RPG command is executed (2).

A similar five-line listing follows (3). However, errors have been imbedded in these lines for the sake of the example. When these lines are compiled, the screen again exhibits the compilation, but the message in the Command Mode window reports on the errors and warnings (4).

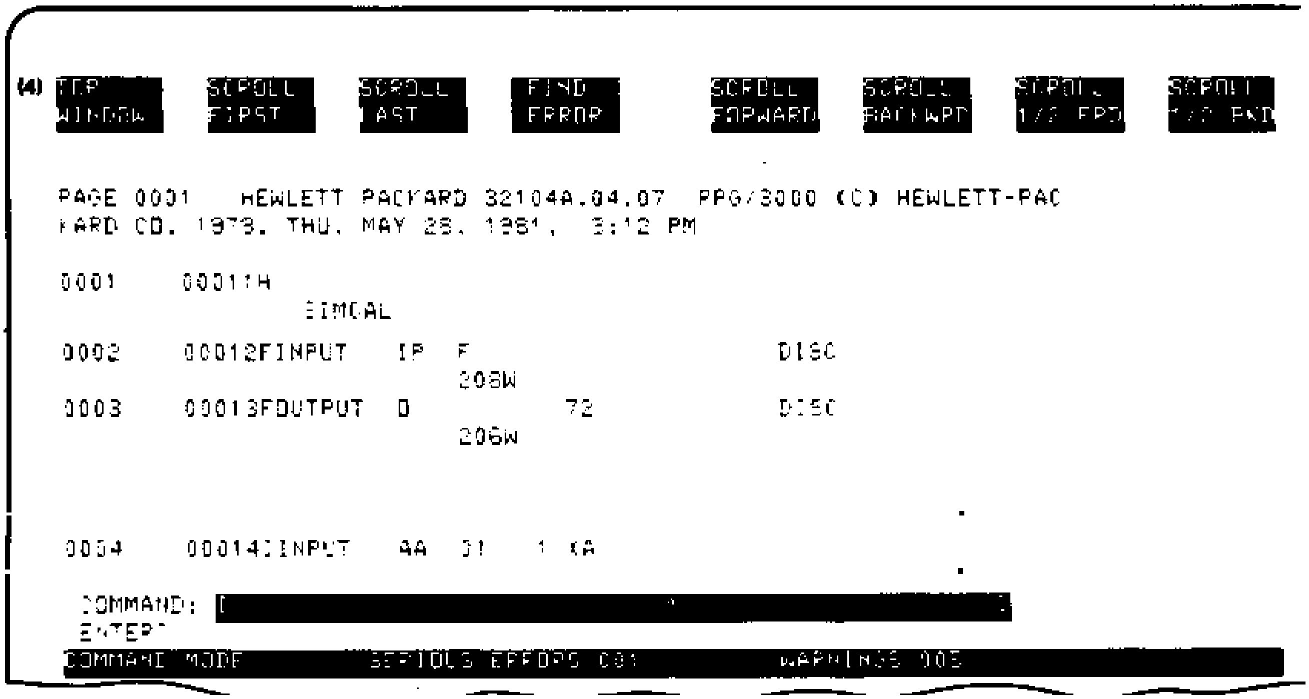
```
>
>L1/5
```

```
         1     00011H
         SIMCAL
         2     00012FINPUT    IP  F                        DISK
         3     00013FOUTPUT   O           72               DISK
         4     00014IINPUT    AA  01   1  CA
         5     00015I        OR   02   1  CS
(1)  >VERIFY RPG
```



```
PAGE 0001   HEWLETT PACKARD 32104A.04.07  RPG/3000 (C) HEWLETT-PAC
KARD CO. 1978. THU, MAY 28, 1981,  3:08 PM

0001     00011H
                SIMCAL

0002     00012FINPUT   IP  F        72              DISC
0003     00013FOUTPUT  O   F        72              DISC



0004     00014IINPUT    AA  01   1 CA
0005     00015I        OR   02   1 CS
0006     00016I        OR   03   1 CM

 COMMAND:
 ENTER^
(2) COMMAND MODE        SERIOUS ERRORS 001        WARNINGS 005
```

```
(3) >L1/5
     1      00011H
     SIMCAL
     2      00012FINPUT   IP  F                     DISC
     3      00013FOUTPUT  O          72             DISC
     4      00014IINPUT   AA  01   1 XA
     5      00015I       OR   02   1 CS
    >VERIFY RPG
```

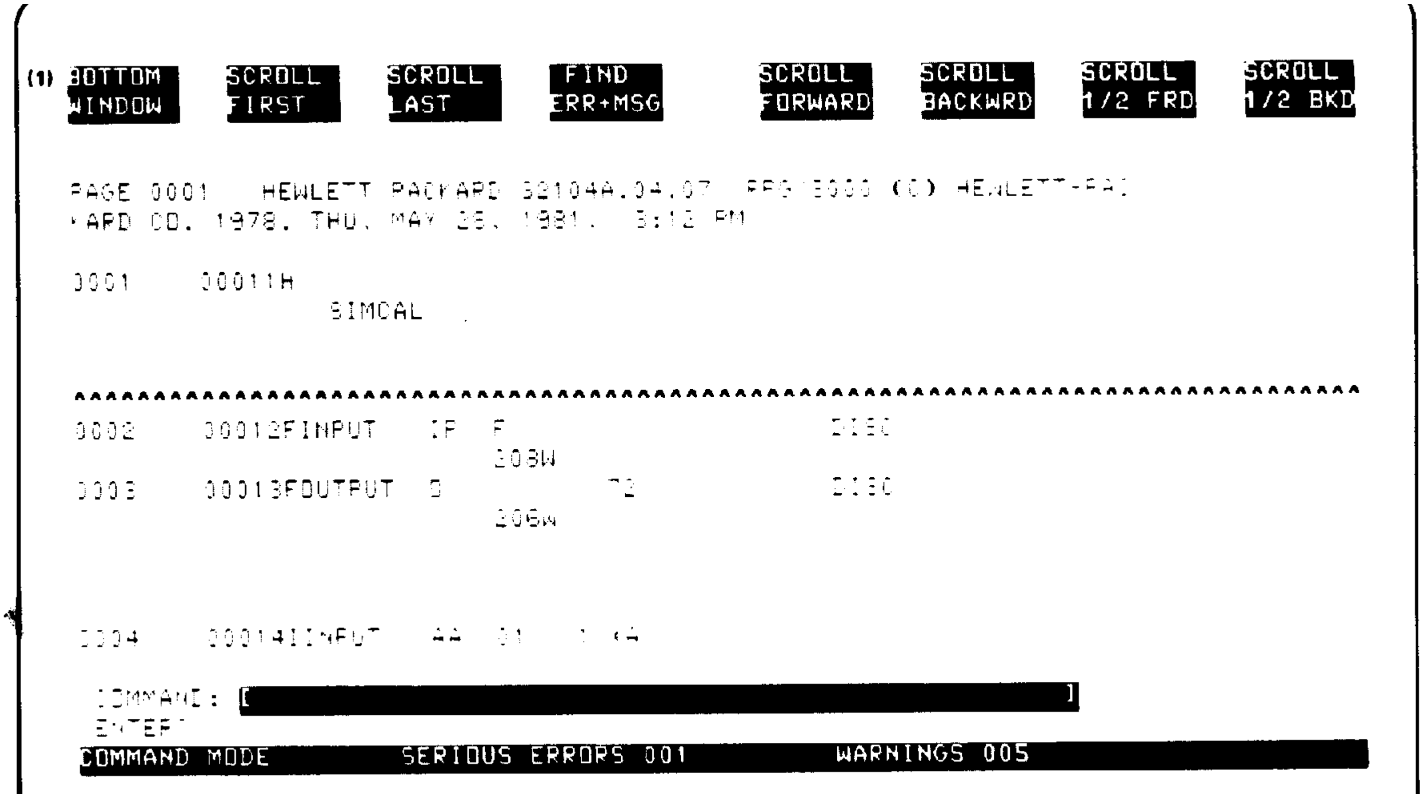


```
PAGE 0001   HEWLETT PACKARD 32104A.04.07  RPG/3000 (C) HEWLETT-PAC
KARD CO. 1978. THU, MAY 28, 1981,  3:12 PM

0001     00011H
                SIMCAL

0002     00012FINPUT   IP  F                    DISC
                           208W
0003     00013FOUTPUT  O          72            DISC
                           206W



0004     00014IINPUT    AA  01   1 XA

 COMMAND:
 ENTER^
COMMAND MODE        SERIOUS ERRORS 001        WARNINGS 005
```

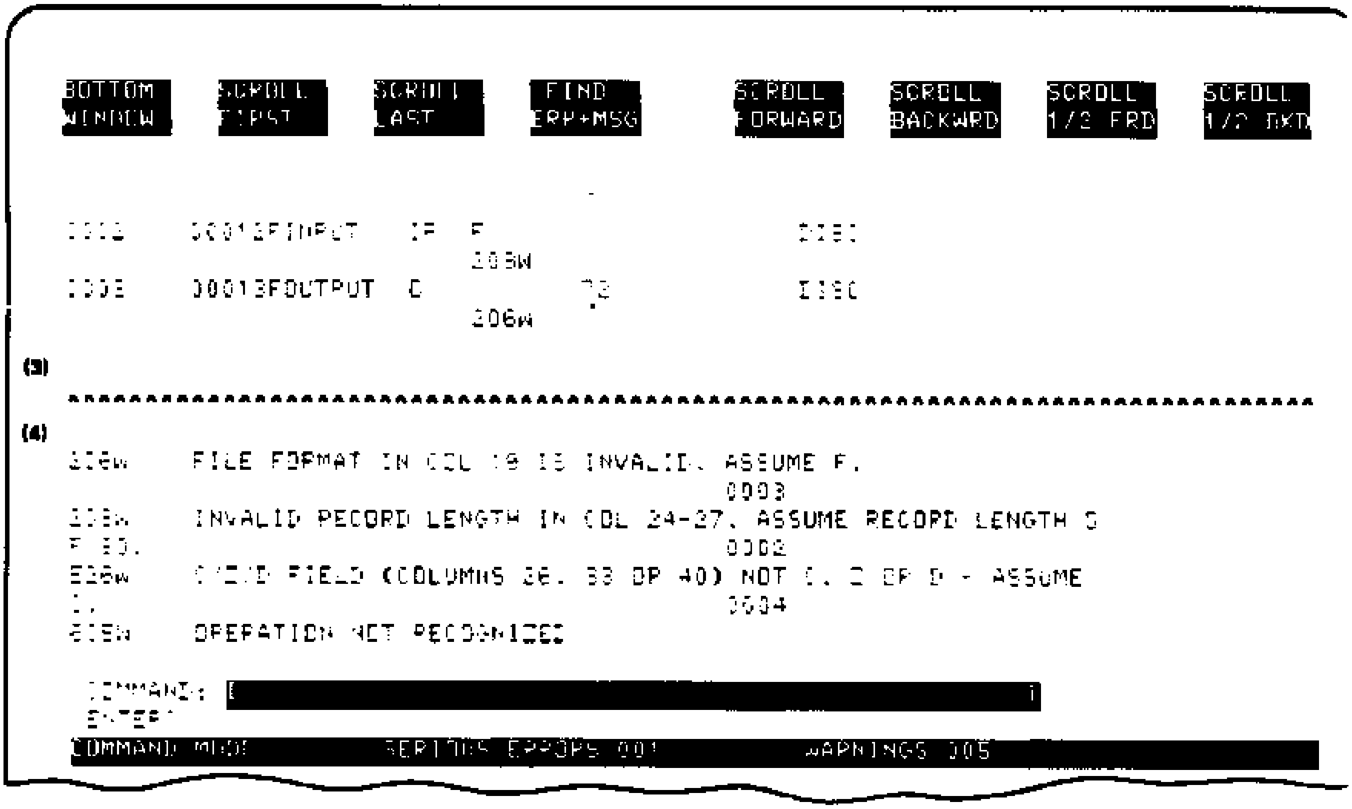
The next example follows the previous one directly. Special function key #1 (TOP WINDOW) has been pressed, and the x screen is split (1). In the bottom half of the screen, error and warning message numbers appear beneath the lines containing the errors (2). Now, if special function key #4 (FIND ERR+MSG) is pressed, the usefulness of the split screen is demonstrated. In the second display, the top portion of the screen contains the compiled source lines and the error and warning numbers (3). The bottom window contains the numbers and their messages (4). Note that the up arrows (^^) indicate that the screen is presently in top window mode.

In the next display, the same information is shown on the screen, but it is now in bottom window mode, as the down arrows (vv) indicate. Pressing F1 caused the shift to bottom window mode (5).

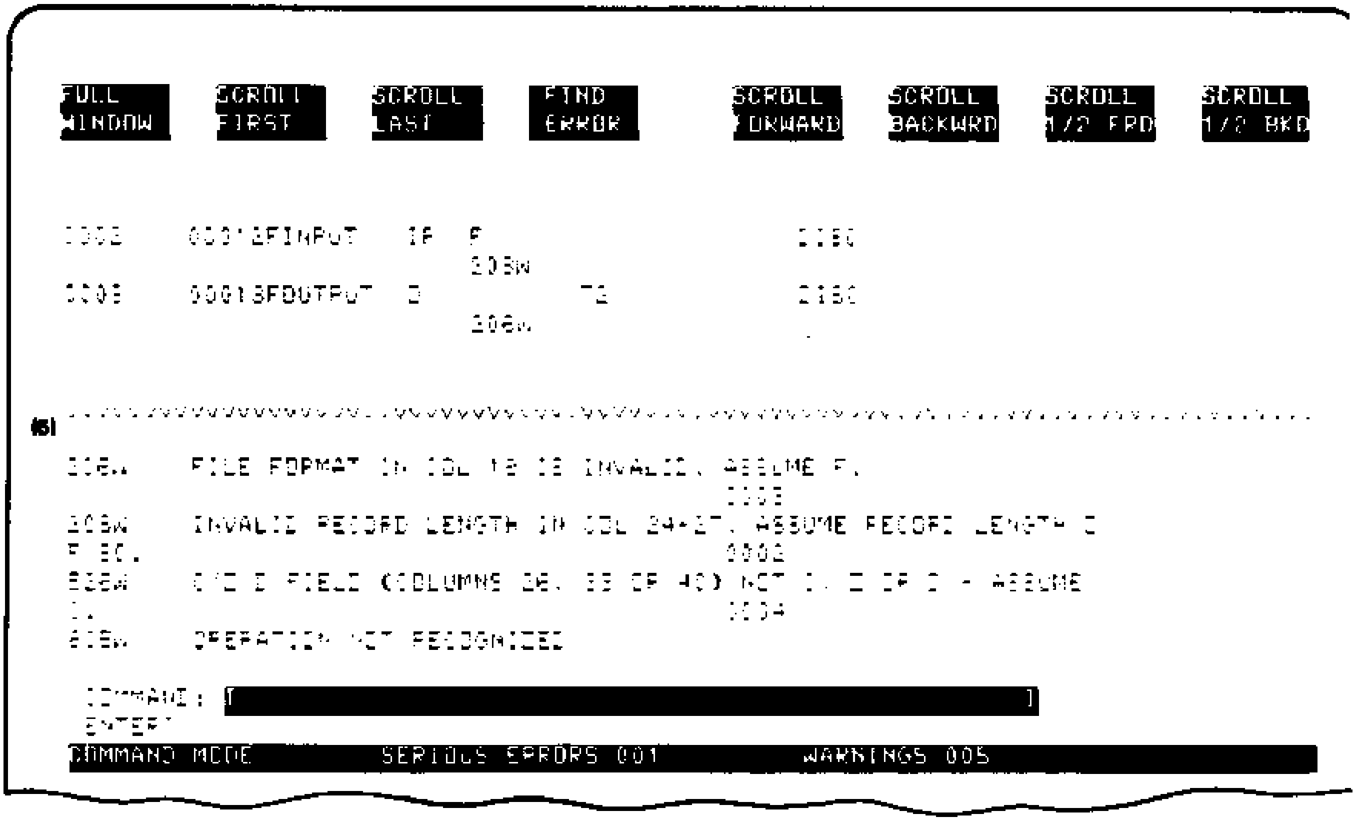
(1)



```
PAGE 0001   HEWLETT PACKARD 32104A.04.07  FRG 3000 (C) HEWLETT-PAC
KARD CO. 1978, THU, MAY 28, 1981,  3:12 PM

0001     00011H
                   SIMCAL

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
0002     00012FINPUT   IP  F                       DISC
                           308W
0003     00013FOUTPUT  O            T2             DISC
                           306W



0004     00014IINPUT   AA  01    1 4A

 COMMAND: [                                                  ]
 ENTER
COMMAND MODE          SERIOUS ERRORS 001           WARNINGS 005
```



```
0002    00013FINPUT   IP  F                      DISC
                          203W
0003    00013FOUTPUT  O           72             DISC
                          206W

************************************************************************
206W    FILE FORMAT IN COL 19 IS INVALID. ASSUME F.
                                                0003
203W    INVALID RECORD LENGTH IN COL 24-27. ASSUME RECORD LENGTH O
F 80.
                                                0002
226W    O/I/D FIELD (COLUMNS 26, 33 OR 40) NOT I, O OR D - ASSUME
I.
                                                0004
605W    OPERATION NOT RECOGNIZED

  COMMAND: [                                                        ]
  ENTER
COMMAND MODE          SERIOUS ERRORS 001          WARNINGS 005
```



```
0002    00013FINPUT   IP  F                      DISC
                          203W
0003    00013FOUTPUT  O           72             DISC
                          206W

........................................................................
206W    FILE FORMAT IN COL 19 IS INVALID. ASSUME F.
                                                0003
203W    INVALID RECORD LENGTH IN COL 24-27. ASSUME RECORD LENGTH O
F 80.
                                                0002
226W    O/I/D FIELD (COLUMNS 26, 33 OR 40) NOT I, O OR D - ASSUME
I.
                                                0004
605W    OPERATION NOT RECOGNIZED

  COMMAND: [                                                        ]
  ENTER
COMMAND MODE          SERIOUS ERRORS 001          WARNINGS 005
```

In the next example, it is assumed that no serious compilation error occurred so the user proceeds to prepare and execute. The VERIFY PREP option is employed, and the Segmenter banner is displayed (1). Next, the MPE command :SAVE assigns a new file designator to $OLDPASS when it is made permanent (2). A second MPE command, :LISTF, with the 2 parameter, brings to the screen file information including file name, file code, record size, record format, data format (3). The input/output file equations are given (4), and the RUN command is issued using the new file designator (5). Again, the calculator is applied (6),nn and a colon, entered by the user (7), returns the user to RISE (8).

```
(1)  >VERIFY PREP
     HP32050A.01.03 SEGMENTER (C) HEWLETT-PACKARD CO. 1979
     ---
      Segmenter completed.

(2)  >:SAVE $OLDPASS,SIMPROG
     >:LISTF SIMPROG,2
     ACCOUNT=  SUBSYS      GROUP= RPG

     FILENAME  CODE ------------LOGICAL RECORD---------  ----SPACE----
                      SIZE  TYP       EOF    LIMIT R/B  SECTORS #X MX

(3)  SIMPROG   PROG    128W  FB        12      12   1      13    1  1

(4)  >:FILE INPUT=$STDIN
     >:FILE OUTPUT=$STDLIST

(5)  >RUN SIMPROG

           SIMPLE           CALCULATOR

(6)  ENTER DATA IN THE FORMAT:
     C NNNDD NNNDD
     A 00111 00222

..... YOUR PROBLEM:    1.11 ADD          2.22 EQUALS      3.3300

     ENTER DATA IN THE FORMAT:

(7)  C NNNDD NNNDD
     :

(8)  HP#####X.00.00 RPG INTERACTIVE SYSTEM ENVIRONMENT RISE
     (C) HEWLETT-PACKARD CO. 1981      THU, JUN 4, 1981, 2:59 PM

     >
```

In the final example, the VERIFY PREP option is demonstrated again, this time with two additional parameters, PMAP and MAXDATA. The PMAP parameter produces a descriptive listing of the file on the screen (1). Following the listing, an execution message informs the user that the segmenting is completed (2). The temporary file is made permanent with the use of the MPE :SAVE command (3), and :LISTF,2 again brings descriptive file information to the screen (4). PROGFILE is then RUN, and the calculator is displayed and applied (5).

```
     >V P PROGFILE;PMAP;MAXDATA=10000
(1)  HP32050A.01.03 SEGMENTER  (C) HEWLETT-PACKARD CO. 1979
     ---
     PROGRAM FILE PROGFILE.RPG.SUBSYS

     RPG'01            0
        NAME            STT  CODE ENTRY SEG
        RPG'OB            1     0     0
        R'OUT             2               1
        R'INT             3               1
        R'CALT            4               1
        R'CALD            5               1
        R'CNTL            6               ?
        SEGMENT LENGTH         34
     RPG'00            1
        NAME            STT  CODE ENTRY SEG
        R'OUT             1     0     0
        R'EDIT            6               ?
        R'LINEC           7               ?
        R'CALT            2   574   574
        R'CALD            3   600   600
        DIVD             10               ?
        R'INT             4   766   766
        R'ERROR          11               ?
        I'0001            5   766  1052
        SEGMENT LENGTH       1130

 PRIMARY DB      134    INITIAL STACK   1440    CAPABILITY      600
 SECONDARY DB    277    INITIAL DL         0    TOTAL CODE     1164
 TOTAL DB        433    MAXIMUM DATA   23420    TOTAL RECORDS    14
 ELAPSED TIME   00:00:04.673             PROCESSOR TIME   00:00.576
 -

(2)   Segmenter completed.

(3)  >:SAVE PROGFILE

(4)  >:LISTF PROGFILE,2
     ACCOUNT=  SUBSYS      GROUP= RPG

     FILENAME CODE  ---------LOGICAL RECORD---------   ----SPACE----
                    SIZE TYP        EOF    LIMIT R/B   SECTORS #X MX

      PROGFILE PROG 128W FB          12       12   1        13  1  1

      >RUN PROGFILE

(5)       SIMPLE           CALCULATOR

      ENTER DATA IN THE FORMAT:
      C NNNDD NNNDD
      S 22222 11111

..... YOUR PROBLEM:  222.22  SUB 111.11 EQUALS    111.1100

      ENTER DATA IN THE FORMAT:
      C NNNDD NNNDD
      :

      HP#####X.00.00  RPG INTERACTIVE SYSTEM ENVIRONMENT  RISE
      (C) HEWLETT-PACKARD CO. 1981       THU, JUN 4, 1981,  3:02 PM

      >
```

XPAND

XPAND expands file limit size of the current work file.

Form

```
X[PAND] numrecs
```

Parameter Explanation

numrecs Number of records by which you wish to expand the file size.

Purpose

The XPAND command increases the file limit size of the current work file by a specified number of records.

Related Commands

When you use the FILE command to create a new file, RISE asks you for the maximum number of records the file will contain. If, for example, you specified 10 lines for the new file, entered 10, and discovered that you had more lines to add to the file, RISE would return the following message:

```
Error 147:  No room in edit file; use XPAND.
```

Using X[PAND] with the number of records you wish to add, you could enlarge the size of the file as desired. RISE would respond with the following message, when the expansion is accomplished:

```
Xpand completed.
```

```
Execution mode:  Line.
```

Examples

The following is a legal abbreviation which may be used with the XPAND command and its parameters:

```
X100 Increase the file limit size of the current work file by 100 records.
```

In the example, a KSAM file is created with a maximum number of 3 records (1). An attempt to LIST the complete contents of the file results in an error message because, at this point, the file is empty (2). A second error is caused by an attempt to enter more records into the file than were specified as the maximum number previously (3). The XPAND command is used to expand the maximum number of records the file will accept by ten (4). When the ADD command is again employed, the file accepts the additional records (5).

```
     >FILE NEWSIM
(1)   Creating new Ksam file, enter maximum number of records.
     3
      File opened.
     >L ALL

(2)   Error 81: Range is empty.
     >A
         1    :LINE 1
         2    :LINE 2
         3    :LINE 3
         4    :LINE 4

(3)   Error 147: No room in edit file, use XPAND.
     >L A
         1     LINE 1
         2     LINE 2
         3     LINE 3
     >

(4)  >XPAND10
      Xpand completed.

(5)  >ADD LAST
         3     LINE 3
         4    :LINE 4
         5    :LINE 5
         6    :<Control Y>

     >
```

:MPEcommand

:MPEcommand executes MPE command.

Form

```
:MPEcommand
```

Purpose

Executes an "MPEcommand" following a colon--:--that can be executed programmatically with RISE. How these commands are executed is described in the MPE Intrinsic Manual under Command Intrinsic.

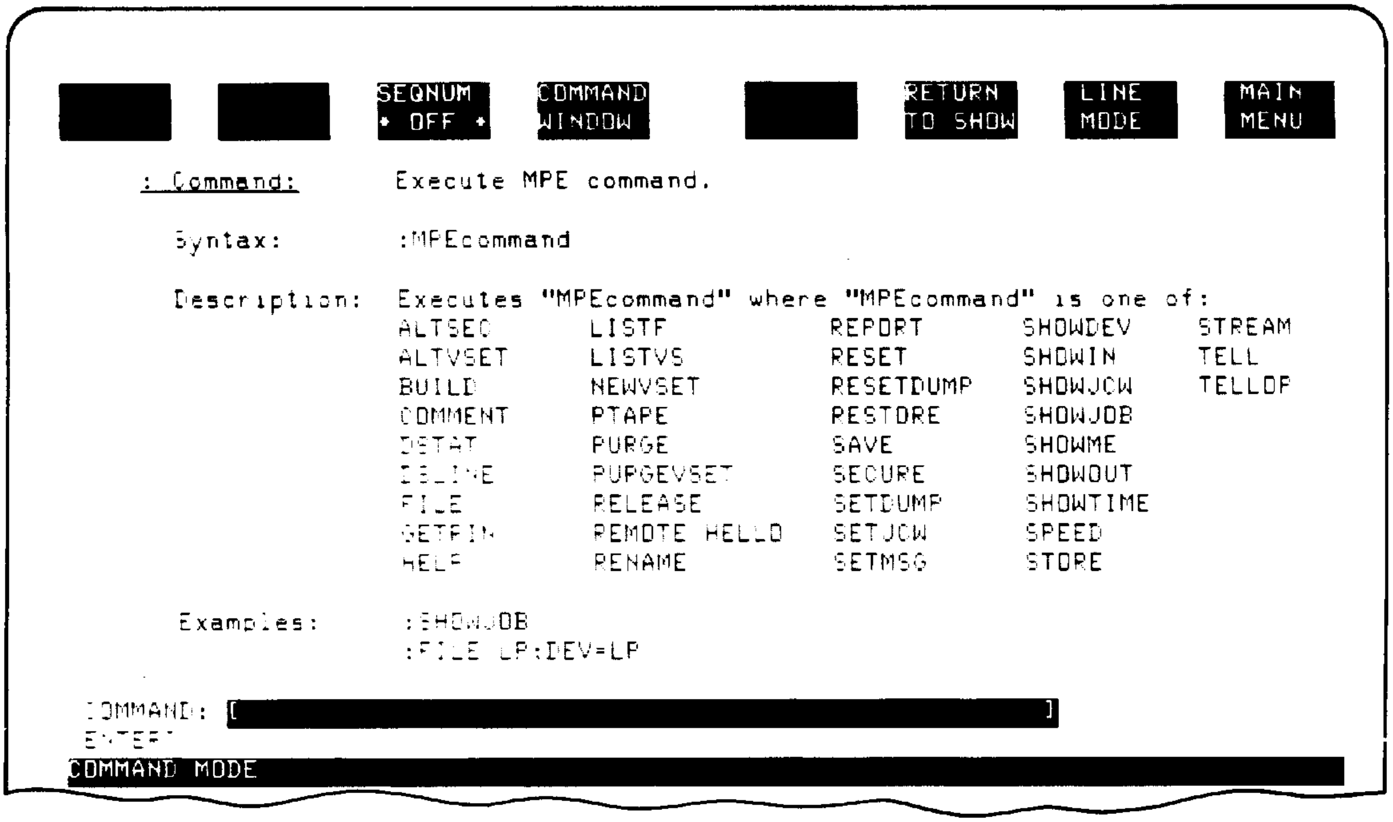
Related Commands

When you type H[ELP]: a list of the MPE commands which can be executed programmatically by RISE will be displayed on the screen.

```
Execution mode:  Line.
```

Examples

A list of some MPE commands that may be executed through RISE will appear on the screen when HELP: is entered. Below is a copy of the command display which appears in response to this command.



```
    : Command:      Execute MPE command.

      Syntax:       :MPEcommand

      Description:  Executes "MPEcommand" where "MPEcommand" is one of:
                    ALTSEC      LISTF          REPORT      SHOWDEV    STREAM
                    ALTVSET     LISTVS         RESET       SHOWIN     TELL
                    BUILD       NEWVSET        RESETDUMP   SHOWJCW    TELLOP
                    COMMENT     PTAPE          RESTORE     SHOWJOB
                    DSTAT       PURGE          SAVE        SHOWME
                    DSLINE      PURGEVSET      SECURE      SHOWOUT
                    FILE        RELEASE        SETDUMP     SHOWTIME
                    GETRIN      REMOTE HELLO   SETJCW      SPEED
                    HELP        RENAME         SETMSG      STORE

      Examples:     :SHOWJOB
                    :FILE LP;DEV=LP

 COMMAND: [                                                    ]
 ENTER
COMMAND MODE
```

In this example, a file equation is entered which specifies file L as a line printer (1). The results of entering :SHOWTIME follow (2). A LISTF,2 provides you with information about the current work file (3). A means of purging a file ends the example (4). Note that, in all the examples, the user provided the colon (:) prompt.

```
(1)  >:FILE L;DEV=LP

(2)  >:SHOWTIME
```

```
     THU, MAY 28, 1981, 3:20 PM
     >

(3)  >:LISTF BACKUP,2
     ACCOUNT=   SUBSYS    GROUP= RPG

     FILENAME   CODE  -------LOGICAL RECORD----------   ----SPACE----
                      SIZE TYP      EOF     LIMIT R/B   SECTORS #X MX

     BACKUP            88B FA       35         38   2        20  7  7

(4)  >:PURGE BACKUP
     >
     >
```

11 RECOVERY PROCEDURES

While using the RPG Interactive System Environment, you may encounter one of six problem situations. They are:

- System failures.
- Terminal lockouts.
- Breaking out of Block Mode.
- Transmission errors.
- Loss of echo at the screen.
- Effects of :EOD.

For your information, RISE uses the following file designators during operation:

- *Wdddhhmm* --KSAM data file for work file.
- *WKddhhmm* --KSAM key file for work file.
- *Ldddhhmm* --compilation listing file.
- *Udddhhmm* --Undelete file (in temporary domain).

If RISE abnormally terminates, because of system failure, use of the Break key, or entering :EOD, you may find these files are permanent in your group and account. The KSAM work file should be recovered, and the compilation listing file and the UNDELete file can be purged.

Recovering from System Failures

If the system fails while you are editing a copy of a file (initiated by the TEXT command), do the following:

1. Determine the designator of the file or files placed in jeopardy by the failure.

   a. The name of the work file, which is a permanent KSAM file, is in the form Wdddhhmm, where "ddd" is the day of the year, "hh" is the hour of the day, and "mm" is the minute of the hour.

   b. The name of the associated key file is WKddhhmm, where K overlays the first "d" in day of the year, and "ddhhmm" have the same meaning as they do with the work file.

   c. Perform a:LISTFW@######,2 to determine the designators of the files.

2. After you have the name of the work file, enter the following command:

   ```
   :RUN KSAMUTIL.PUB.SYS
   ```

3. When you receive a ">" prompt at the terminal enter the following two commands to KSAMUTIL:

```
        >KEYINFO filename;RECOVER

        (and, after recovery is made)

        >EXIT
```

(In the above, "filename" is the name of the KSAM data file determined in the LISTF command.)

This will recover the work file and take you out of KSAMUTIL. You can then run RISE and continue editing after you TEXT in the recovered work file name.

If the system fails while you are editing the file directly (initiated by the FILE command), perform steps 2 and 3 above. In other words, use KSAMUTIL and the KEYINFO command with the RECOVER option with the name of the direct file to restore it for continued editing.

Recovery from Terminal Lockout

Terminal lockout is the term used to describe a state in which the terminal is totally unresponsive to any key you press. To recover, do the following:

```
  1.  If you are in Line Mode, just press the RESET TERMINAL key twice.
      This is a hard reset.

  2.  If you are in Block Mode, press the RESET TERMINAL key twice, for
      a hard reset.  Then press softkey F7 to enter Line Mode.  An
      exception occurs when you are in Scroll set.  If you are in Scroll
      set (SHOW mode), the special function key F7 has a different
      function.  In order to return to Line Mode, you must first press
      F1, which will take you out of Scroll set and into Master set.
      You may then press F7 to enter Line Mode.
```

Recovery from Breaking out of Block Mode

If you hit the BREAK key inadvertently while in Block Mode, do the following:

```
  1.  Do a hard reset (press the RESET TERMINAL key twice).

  2.  Type "ESC:" to turn echo on.  (At this point you should be able to
      communicate with MPE.)

  3.  Type RESUME to resume execution of the editor.  MPE will respond
      with "Read Pending."

  4.  Enter the carriage return.

  5.  If you are in Scroll set (SHOW mode), press F1 to reset the
      special function keys to Master set.

  6.  Press softkey F8 to display the main menu.

  7.  Press softkey F8 to refresh the main menu.  This will return you
      to Block Mode.
```

NOTE Steps 6 and 7 may be skipped if you press softkey F7, which puts you in Line Mode instead of Block Mode.

Recovery from a Transmission Error

Transmission errors usually bring to the screen a nonsensical clutter of characters. The keyboard works but characters, such as escape characters, appear on the screen erroneously. They cause incorrect displays of forms in block mode.

To recover from the effects of transmission errors do the following:

1. If you are in Scroll set (SHOW mode), press F1 to reset the special function keys to normal SHOW mode functions (Master set).
2. Press F8 to display the main menu.
3. Press F8 a second time to refresh the main menu.

You can also press F7 to enter Line Mode and reenter your command.

Recovering Echo After It Is Lost

When your screen remains blank everytime you type a key, your terminal may have lost its echo. You can recover it in one easy step.

1. Press the ESCAPE key followed by the colon (:).

Recovery from :EOD

If you accidentally or intentionally enter :EOD (End of Data) while in Line mode, RISE will immediately terminate. Your KSAM work file will not be purged.

If you were editing a copy of a permanent file (TEXT) or had created a new work file (ADD), you can locate the work file by typing the following:

```
:LISTF W@######,2
```

See section A, above, for a discussion of this file designator.

To recover this work file, you simply TEXT in the KSAM data file and KEEP it to a sequential permanent file. Then you use the KSAMUTIL to purge the KSAM work file with the name located using :LISTF as described above.

If you entered :EOD while editing a file directly (FILE), no harm is done, and the :EOD operates like the EXIT command. Your changes to the file will be permanent.

E Appendix A COMMAND SUMMARY

3 A[DD] [lb] [BY inc] [^^[form] [R[EPEAT] [n]]]

ADD lines starting at "lb" BY inc (rement); "^^" controls the RPG Record Spec forms and may be entered while in Add Mode at the start of a new line; the "R[EPEAT] n" option homes the cursor and clears the screen after "n" number of lines have been added. The LIST, DELETE, and MODIFY commands may be entered while in the Add Mode by typing ">>" at the start of a new line followed by the command.

Also, you can ">>" alone and RISE will prompt you for text starting at column 6 instead of column 1. The command automatically skips over the RPG line numbers in column 1-5. You type "<<" to get back to column 1.

4 B[EGIN] [N[OW]]

End current editing session to start editing a new file as if you had just entered RISE; if editing session was initiated by FILE command, close the direct work file to keep the changes; if editing session was initiated by TEXT command, display safety messages before purging the work file; the N[OW] option disables the safety messages and clears the file, initiated by the TEXT command, immediately.

5 CH[ANGE] "oldstring" TO "newstring" [IN lb[/le]]

Changes "oldstring" to "newstring", starting at "lb" and ending at "le".

6 COM[MENT] [C] [lb] [BY inc]

Displays a comment banner 10 lines long with the RPG Record Spec form of the type corresponding to the line before "lb", where the banner will appear; "C" increases the banner size to 15 lines and displays a column indicator; "BY inc" establishes the increment value for each line within the banner.

7 COP[Y] lb[/le] TO loc [BY inc]

Duplicates lines starting at "lb" TO a specified "loc"ation; "lb/le" establishes a range of lines to be copied; "BY inc" sets the increment value for each line being copied.

8 D[ELETE] lb[/le]

Delete line or lines starting at "lb"; stop the deletion after "le".

9 E[XIT] [N[OW]]

Ends the subsystem; "NOW" parameter establishes that the command is to be executed immediately, without RISE prompting you with safety messages and without saving the work file with the KEEP command.

10 FIL[E] filename [N[OW]]

Allows you to edit a file directly without the delays resulting from the TEXT/KEEP commands; the "NOW" parameter used with this command disables safety prompts and will open "filename" even if you have been editing a work file that was created with the TEXT command and you did not keep your changes in a permanent file. The "NOW" parameter immediately discards the current work file.

11 F[IND] [E[VERY]] "string" [IN lb[/le]]

Locates "string"; "EVERY" specifies that every occurrence of "string" is to be located; if "EVERY" is omitted, only the first occurrence is located; you type "F" alone to find the next occurrence; "IN lb/le" restricts the search to a range.

12 FO[RM] form

Displays the RPG Record Spec form of type "form". "form" may be H,F,E,L,I,C,O, or COL.

13 G[ET] filename

Reads and executes commands stored in the file "filename".

14 H[ELP] [command]

Gives brief desciptions of all RISE commands; when "command" is specified, provides detailed descriptions of single command.

15 I[NCR] [value]

Sets default increment value; "value" sets new default increment value which prevails until "I[NCR] " is used again.

16 J[OIN] filename [TO loc] [BY inc]

Appends or merges an entire file named "filename"; "TO loc" locates position where "filename" is to be joined; "BY inc" establishes increment value.

17 K[EEP] [filename] [N[OW]] [U[NN]]

Copies the work file into a file named "filename"; "NOW" parameter instructs RISE to overwrite an existing copy of the file without RISE prompting you with safety messages; "UNN" specifies that the file is kept without appended sequence numbers.

18 LIN[E]

Enters line mode.

19 L[IST] [lb[/le]] [O[FF]]

Lists lines in file; "le/lb" is the range of lines for listing "OFF" lists the lines offline to a line printer instead of to your terminal with the formal file name of "RISELIST" and device class of "LP".

20 ME[NU]

Displays main menu.

21 M[ODIFY] [lb[/le]] [^^]

Lists line[s] for modification; "lb/le" establishes range of lines for modification; "^^" causes appropriate RPG Record Spec form to appear with the line. (Type <Control Y> to restore record and "//" to exit a range of lines.)

22 MOV[E] lb[/le] TO loc [BY inc]

Moves line (or lines) "lb/le" to a specified location--"TO loc"; "BY inc" parameter establishes increment values.

23 P[RINT] lb[/le]] [R[PGLIST]]

Prints line (or lines) offline in formatted form, skipping 3 lines whenever the form type in column 6 changes, with the formal filename of "RISELIST" and device class of "LP"; if "RPGLIST" is specified, the compilation listing is printed offline, instead of the work file, with the formal filename of "RPGLIST".

24 R[ENUM] [lb/le] [BY inc]

BOTH [BY inc]

SO[URCE] [BY inc]

SO[URCE] W[ITH] SE[QNUM]

SE[QNUM] W[ITH] SO[URCE]

Renumbers work file sequence line numbers; "lb/le" establishes range for renumbering; "BY inc" establishes increment value; "BOTH" renumbers sequence line numbers and RPG source line numbers (cols 1- 5); "SOURCE WITH SEQNUM" renumbers source lines to agree with the sequence line numbers; "SEQNUM WITH SOURCE" renumbers sequence line numbers to agree with source line numbers.

25 RU[N] progfile [,entrypoint] [;NOPRIV] [;LMAP] [;DEBUG]

[;MAXDATA=segsize] [;PARM=num]

[;STACK=stacksize] [;DL=dlsize]

P

[;LIB=G] [;NOCB]

S

Runs a program file (similar to MPE:RUN command); see RUN command for an explanation of parameters.

26 24. S[HOW] [lb] [C[OL]]

Shows a page for direct screen modification with RPG Record Spec

form appropriate to records displayed; "COL" parameter displays a column indicator rather than the RPG Record Spec form.

27 T[EXT] filename [N[OW]]

Copies the file "filename" into a work file; "NOW" allows you to TEXT in a new work file without keeping the changes you have made

to the work file you are editing and without RISE prompting you with a safety message.

28 UN[DEL]

Restores lines deleted by the last "DELETE" command.

29 V[ERIFY] R[PG]

L[IST)

P[REP] [progfile] [;ZERODB] [;PMAP]

[;MAXDATA=segsize] [;STACK=stacksize]

[;DL=dlsize] [;CAP=caplist] [;RL=filename]

[;PATCH=patchsize]

"RPG" compile the edit file; or "LIST" a previously produced compilation listing. "PREP" prepares a USL file in $OLDPASS to a program with the preparation "options" that are described under VERIFY.

30 X[PAND] numrecs

Expands the size of the work file by "numrecs" (number of records).

31 :MPEcommand

Executes an "MPEcommand" entered following a colon (:) character. The MPE commands which you may enter are listed when you type H[ELP]:.

F Appendix B COMPARISON OF RISE AND EDIT/3000

RISE and EDIT/3000 are in many respects similar editors. There are significant differences between them, however. Below is a list of contrasting features that new RISE users, who are accustomed to EDIT/3000, will find helpful.

RISE Features	**EDIT/3000 Features**
1. Line and Block Modes.	Line mode only.
2. No delimiters between parameters.	Delimiters between parameters.
3. Only capital letters accepted.	Capital and lower case letters accepted.
4. KSAM work file.	Extra data segments in addition to a work file.
5. Only 1 command accepted at a time.	Multiple commands allowed.
6. No quiet mode.	Optional quiet.
7. UNDEL command to recover deletions.	No delete recovery.
8. PRINT ALL prints offline with spacing	LIST ALL, OFFLINE prints offline.

between changes in column 6 (form type). LIST ALL OFF prints offline without spacing.	
9. HELP command replaces.	XPLAIN command.
10. MOVE command replaces.	GATHER command.
11. RENUM command replaces.	GATHER ALL command.
12. GET command replaces.	USE command.
13. Command must be completed on 1 line (can't exceed 50 characters in length).	Ampersand (&) used to continue a command on next line.

G Appendix C INFORMATIVE EXECUTION MESSAGES

NUM	Informative Execution Message
0	Syntax checking is turned on.
1	Syntax checking is turned off.
2	Preparing for compilation...
3	Return to main menu with the MENU command, then compile.
4	Text completed.
5	Warning: record size is greater than 88 bytes, truncated to 80.
6	File is unnumbered; appending sequence numbers.
7	Print completed.
8	Increment reset to 1.
9	Increment set.
10	No errors found.
11	At End-Of-File.
12	No compilation errors occurred.
13	Error message not found.
14	Position cursor at line to add after, then press softkey F1-F4.
15	Insert lines cancelled.
16	No lines to show; displaying main menu.
17	Keep completed.
18	Delete completed.
19	Can't Un-delete a line; the line number exists already.
20	Undel completed.
21	'String' not found.
22	String found.
23	Changes completed.
24	No changes were made.
25	Entering Line mode - use Carriage Return.
26	Entering Block Mode - use ENTER key.
27	TITLE.

NUM	Informative Execution Message
28	Xpand completed.
29	Creating new Ksam file; enter maximum number of records.
30	Enter response and press ENTER key.
31	Errors in response; Command cancelled.
32	File opened.
33	Enter new file name followed by additional number of records.
34	File exists already; enter a NEW file name.
35	Join completed.
36	Copy completed.
37	Move completed.
38	Renum completed.
39	Get completed.
40	Creating work space file.
41	Keep cancelled.
42	End-Of-Data.
43	KEEP not done. CLEAR current work file?
44	File cancelled.
45	Text cancelled.
46	Exit cancelled.
47	Offline listing completed.
48	Segmenter completed.
49	Ready to begin new session.
50	Begin cancelled.

H Appendix D ERROR MESSAGES

NUM	Error Message
0	Can't sense cursor position.
1	Only one command allowed.
2	Incorrect specification of the command in the menu.
3	Only F,T,K,J allowed in File type command.
4	Only I,P,C,R,M allowed in Gen1 type command.
5	Only D,S,A,F,CO,CH allowed in Edit type command.
6	Only U,H,V,E allowed in Gen2 type command.
7	There is no such editor command.
8	Syntax error on ADD command.
9	Syntax error on CHANGE command.
10	Syntax error on COMMENT command.
11	Syntax error on COPY command.
12	Can't distinguish between COMment or COPy command.
13	Can't distinquish CHange, COMment, or COPy command.
14	Syntax error on DELETE command.
15	Syntax error on EXIT command.
16	Can't recognize the command.
17	Syntax error on FILE command.
18	Syntax error on FIND command.
19	Can't distinquish between FILe, FINd, or FOrm.
20	Syntax error on HELP command.
21	Syntax error on INCR command.
22	Syntax error on JOIN command.
23	Syntax error on KEEP command.
24	Syntax error on MENU command.
25	Syntax error on MOVE command.
26	Can't distinquish MEnu, MOVe, or MODify command.
27	Syntax error on PRINT command.
28	Syntax error on RENUM command.

NUM	Error Message
29	Syntax error on SHOW command.
30	Syntax error on TEXT command.
31	Syntax error on UNDEL command.
32	Syntax error on VERIFY command.
33	Syntax error on NOW parameter.
34	Only NOW parameter allowed with Exit command.
35	KEEP not done - use NOW option to CLEAR work file.
36	Illegal special character.
37	
38	Syntax error on RPG option.
39	Syntax error on ON option.
40	Syntax error on OFF option.
41	Can't distinquish between ON or OFf.
42	Only one of RPG, LIST, or PREP allowed.
43	Need Process Handling Capability.
44	There is no file to compile.
45	Unable to create sort process for compilation.
46	Unable to activate RPG compiler.
47	Unable to set up File equations for compilation.
48	Unable to Build RPGLIST file for compilation.
49	Syntax error on LIST option.
50	There is no compilation listing; use the RPG option.
51	Increment too big; line numbers exceed limit 99999.999.
52	There are no records in the file.
53	A numbered file has illegal characters as line numbers.
54	A file name is required with this command.
55	A file name may only contain a max. of 37 characters.
56	Filename must start with letter followed by letter/num.
57	Only NOW option allowed with this command.
58	Can't edit this type of file, only RPG source programs.
59	A line number is required.
60	A line number is required before the RPGLIST option.

NUM	Error Message
61	Only a ‘*',line number,FIRST,LAST,ALL allowed in range.
62	The TO line cannot be less than the FROM line number.
63	Only the RPGLIST option is allowed after the range.
64	There is no compilation listing to print.
65	There is no edit file.
66	Syntax error on Rpglist option.
67	Syntax error on All option.
68	A special delimiter may not follow the line range.
69	Illegal delimiter.
70	Syntax error on First.
71	Syntax error on Last.
72	A maximum of 5 digits is allowed left of decimal.
73	Range is empty.
74	A maximum of 3 digits is allowed right of decimal.
75	A number is required to right or left of decimal.
76	Illegal numeric increment value.
77	Illegal parameters in command.
78	An increment value cannot be zero.
79	Syntax error on COL option.
80	No parameters allowed after COL option.
81	Range is empty.
82	No room in file; use XPAND to expand file.
83	Can't insert a new line.
84	New insert line number exceeds 99999.999.
85	No room between lines to insert new line number.
86	Cursor must be pointing at a line.
87	Blanks may only follow the UNN option.
88	Syntax error on UNN option.
89	No edit file name for default; give Keep file name.
90	File exists already, use NOW option to replace it.
91	Only NOW and/or UNN option allowed.
92	No lines were Deleted to Un-delete.

NUM	Error Message
93	No parameters are allowed with this command.
94	No string to find.
95	Missing range after IN option.
96	Only a delimited string and/or IN range allowed.
97	Missing close delimiter.
98	A string must be given between the delimiters.
99	A slash (/) cannot delimit a string.
100	Should be blank after line range.
101	‘oldstring' TO ‘newstring' missing.
102	Oldstring & newstring must be enclosed by delimiters.
103	TO missing after ‘oldstring'.
104	Only IN range allowed after ‘newstring'.
105	Missing close delimiter for ‘oldstring'.
106	An oldstring must be specified between delimiters.
107	Missing close delimiter for ‘newstring'.
108	A newstring must be specified between delimiters.
109	Oldstring length must equal newstring length.
110	Syntax error on LINE command.
111	Syntax error on LIST command.
112	Can't distinquish between LINe or LISt command.
113	Missing MPE command.
114	Undefined MPE command.
115	Could not execute this command.
116	Can't keep to a KSAM file&--;only sequential file.
117	Syntax error on RUN command.
118	Can't distinquish between REnum and RUn commands.
119	Missing program name.
120	Unable to create process.
121	Missing secondary entry point after comma.
122	Expecting parameter after semi-colon.
123	Illegal RUN parameter.
124	Can't activate process.

NUM	Error Message
125	Expecting equal sign after a parameter.
126	Illegal numeric value after the equal sign.
127	Numeric value may only have a maximum of 5 digits.
128	Only G,P,S allowed for libsearch value.
129	Program name may contain a maximum of 8 chars.
130	Expecting lockword immediately after the slash.
131	Lockword may contain a maximum of 8 characters.
132	Expecting group name immediately after period.
133	Group name may contain a maximum of 8 chars.
134	Expecting account name immediately after period.
135	Account name may contain a maximum of 8 chars.
136	Syntax error on XPAND command.
137	Missing number of records to expand by.
138	Illegal numeric value.
139	Expansion value must be greater than zero.
140	Can't determine a unique temporary file name.
141	Only BY increment allowed after TO.
142	Only TO line and/or BY increment allowed.
143	Internal error&--;could not convert to double value.
144	No room between line numbers.
145	Missing or illegal line number.
146	Missing numeric increment after By.
147	No room in edit file; use XPAND.
148	Missing line range.
149	TO location missing.
150	Only BY increment allowed after TO location.
151	TO location cannot be within line range.
152	Only BY increment allowed after range.
153	Increment cannot be less than 1 for source.
154	Only BY increment allowed after BOTH.
155	Syntax error on BY or BOTH.
156	BY increment cannot be greater than 5000.

NUM	Error Message
157	Expecting SOURCE after WITH.
158	Expecting WITH SOURCE after SEQNUM.
159	Expecting SEQNUM after WITH.
160	Only WITH or BY allowed after SOURCE.
161	Syntax error on SOURCE.
162	Syntax error on SEQNUM.
163	Can't distinquish between SOurce and SEqnum.
164	Syntax error on WITH.
165	A source line number has non-numeric characters.
166	A source line number is not in ascending sequence.
167	No room between lines to insert a page.
168	Syntax error on Get command.
169	Can't execute Get command from a file.
170	Only 0-8 allowed for a softkey from a file.
171	Pause length must be 0 to 64000.
172	Syntax error on FOrm.
173	Only H,F,E,L,I,C,O, or COI allowed.
174	Missing or illegal offset integer after *.
175	Expecting ^^.
176	Only H,F,E,L,I,C,O,COL and/or Repeat allowed after ^^.
177	Syntax error on By.
178	Syntax error on Repeat.
179	Illegal integer value after Repeat.
180	Unexpected error&--;read from terminal failed!
181	Line truncated to 80 characters.
182	Only DELETE, LIST or MODIFY allowed in Add mode.
183	FATAL ERROR in Show Mode, returning to Line Mode.
184	Line would exceed 80 characters.
185	Syntax error on MODIFY command.
186	Can't distinquish MODify and MOVe commands.
187	A subcommand may only be entered in columns 1-80.
188	Should be blank after Repeat option.

NUM	Error Message
189	Should be blank after BY increment.
190	Should be blank after file name.
191	Should be blank after ^^.
192	Only ^^ allowed after line range.
193	Should be blank after SOURCE.
194	Should be blank after SEQNUM.
195	Only capital letters are allowed.
196	Syntax error on EVERY.
197	Should be blank after OFF.
198	Only OFF allowed after line range.
199	Syntax error on PREP.
200	No USL in $OLDPASS, use RPG option to compile.
201	Could not activate Segmenter.
202	$OLDPASS is only allowable system file name.

""""""""

HP RPG/XL Utilities-Part 3 SIGEDIT

Preface to RPG SIGEDITOR

This is the reference manual for Hewlett-Packard's SIGEDITOR. SIGEDITOR is a menu-driven forms design utility used by RPG programmers to construct forms.

This manual assumes that you are an experienced RPG programmer using SIGEDITOR. In this manual, the term user refers to the person entering data into the forms that you have designed.

This manual contains the following:

Chapter 1	Gives an overview of SIGEDITOR. A short tutorial introduces you to the menu and the basic functions. All topics in this chapter are explained in detail in chapter 2.
Chapter 2	Describes in detail how to use SIGEDITOR and gives examples. This chapter explains forms specifications, fields, and advanced editing techniques.
Chapter 3	Gives step-by-step guides to specific tasks. If you are not familiar with SIGEDITOR, you can use this chapter as a quick guide to these tasks.
Chapter 4	Describes how RPG programmers use forms generated by SIGEDITOR.
Appendix A	Lists and describes the SIGEDITOR error messages.
Appendix B	Summarizes the SIGEDITOR keys.
Appendix C	Describes how to calculate the self-check field.

Additional Documentation

For more information on RPG, see the *HP RPG/XL Reference Manual* and *PROCMON/3000*.

Unsupported Devices

SIGEDITOR does not support these HP terminals:

* 264X Series

* 2621A/B

* 2703A

* 2625A

* 3XXX Series

Conventions

NOTATION DESCRIPTION

UPPERCASE Within syntax statements, characters in uppercase must be entered in exactly the order shown, though you can enter them in either uppercase or lowercase. For example:

SHOWJOB

Valid entries: showjob ShowJob

SHOWJOB

Invalid entries: shojwob ShoJob

SHOW_JOB

italics Within syntax statements, a word in italics represents a formal parameter or argument that you must replace with an actual value. In the following example, you must replace *filename* with the name of the file you want to release:

RELEASE *filename*

punctuation Within syntax statements, punctuation characters (other than brackets, braces, vertical parallel lines, and ellipses) must be entered exactly as shown.

{ } Within syntax statements, braces enclose required elements. When several elements within braces are stacked, you must select one. In the following example,

you must select ON or OFF:

{ON }

SETMSG {OFF}

[ ] Within syntax statements, brackets enclose optional elements. In the following example, brackets around ,TEMP indicate that the parameter and its delimiter are optional:

PURGE {*filename* } [,TEMP]

When several elements with brackets are stacked, you can select any one of the elements or none. In the following example, you can select devicename or deviceclass or neither:

SHOWDEV [*devicename*]

[*deviceclass*]

[...] Within syntax statements, a horizontal ellipsis enclosed in brackets indicates that you can repeatedly select elements that appear within the immediately preceding pair of brackets or braces. preceding pair of brackets or braces. In the following example, you can select *itemname* and its delimiter zero or more times. Each instance of itemname must be preceded by a comma:

[,*itemname*][...]

If a punctuation character precedes the ellipsis, you must use that character as a delimiter to separate repeated elements. However, if you select only one element, the delimiter is not required. In the following example, the comma cannot precede the first instance of itemname:

[*itemname*][,...]

|...| Within syntax statements, a horizontal ellipsis enclosed in parallel vertical lines indicates that you can select more than one element that appears within the immediately preceding pair of brackets or braces. However, each element can be selected only one time. In the following example, you must select ,A or ,B or ,A,B or ,B,A :

{,A}
{,B}|...|

If a punctuation character precedes the ellipsis, you must use that character as a delimiter to separate repeated elements. However, if you select only one element, the delimiter is not required. In the following example, you must select A or B or AB or BA. The first element cannot be preceded by a comma:

{A}
{B}|,...|

... Within examples, horizontal or vertical ellipses indicate where portions of the example are omitted.

Å Within syntax statements, the space symbol Å shows a required blank. In the following example, you must separate modifier and variable with a blank:

SET[(*modifier*)]Å(*variable*);

underlining User input is underlined. For example:

PROMPT?response

In a syntax statement, brackets, braces or ellipses are underlined if you must enter them. For example:

COMMAND
[[*ParameterA*]] = *ParameterB*

shading Within an example of interactive dialog, shaded characters indicate user input or responses to prompts. In the following example, OMEGA is the user's response to the NEW NAME prompt:

NEW NAME? OMEGA

The symbol indicates a key on the terminal's keyboard. For example, CTRL indicates the Control key.

CTRL*char* CTRL*char* indicates a control character. For example, CTRLY means you have to simultaneously press the Control key and the Y key on the keyboard.

base prefixes The prefixes %,#, and $ specify the numerical base of the value that follows

%*num* specifies an octal number.

#*num* specifies a decimal number.

$*num* specifies a hexadecimal number.

When no base is specified, decimal is assumed.

12 Chapter 1 Introduction to SIGEDITOR

SIGEDITOR is a menu-driven utility that lets you design forms for RPG applications. You design the forms through the RPG Screen Interface (RSI). This chapter introduces SIGEDITOR with a short tutorial showing you how to create and modify a form using basic SIGEDITOR functions. For detailed information on using SIGEDITOR, refer to chapter 2.

SIGEDITOR Capabilities

- Create and store up to 1,024 forms in a forms file.
- Modify existing forms.
- Convert S, D, and H RPG specifications from an IBM System 34 or 36 into a format that is compatible with HP RPG programs.
- Print hard copies of forms and form specifications.
- Copy existing forms.
- Delete forms.
- Merge messages from a message catalog file.

SIGEDITOR Menus

- Main menu.
- Edit menu.
- Utilities menu.
- Help menu.

The main menu connects all parts of SIGEDITOR; that is, through the main menu, you can access the other menus. The edit menu allows you to create and alter forms. The utilities menu contains helpful functions, such as deleting a form from a forms file or copying a form from one forms file to another.

Terminology Used in This Manual

user The person entering data into the forms you have designed.

form The specific display into which a user enters data or from which a program displays data. Forms consist of screen constants (such as labels or directions for using the form) and fields.

fields The areas in the form that display or accept data.

designing forms The construction and maintenance of forms.

forms file The file where a collection of related forms are stored. There is a different filename for each collection. A form cannot be created unless it belongs to a forms file.

Tutorial

This short tutorial is designed to teach you some of the basic functions of SIGEDITOR. This lesson guides you through creating a forms file, designing a form, and editing the form.

Running SIGEDITOR

To begin using SIGEDITOR, enter the following command:

```
          :run sigedit.pub.sys
```

The main menu heading appears at the top of the as shown in Figure 1-1.

```
_________________________________________________________________________________
|                                                                               |
|     HP32104A.08.00   SIGEDITOR                         WED, MAR 3, 1988 9:32 AM|
|                                                                               |
|                                                                               |
|                                                                               |
_________________________________________________________________________________
```

Figure 1-1. Main Menu Heading

The main menu keys appear at the bottom of the screen. The keys correspond with f1 through f8 on the keyboard. Figure 1-2 shows the main menu keys.

```
____________________________________________________________________________________
|                                                                                  |
|      SELECT    CONVERT             MERGE        PRINT   GEN I/O     FORM      EXIT|
|     FRMS FIL  S+D SPEC           CATALOG       SPECS    SPECS     UTILS           |
|                                                                                  |
____________________________________________________________________________________
```

Figure 1-2. Main Menu Keys

Creating a Forms File and a Form

To create a forms file press the f1 key (SELECT FRMS FIL). The screen displayed is shown in Figure 1-3.

```
___________________________________________________________________________
|                                                                         |
|     Forms File name:                                                    |
|                                                                         |
|                                                                         |
|                                                                         |
|                                                                  ABANDON|
|                                                                         |
|                                                                         |
___________________________________________________________________________
```

Figure 1-3. Form File Creation Screen

SIGEDITOR requests a name for the new forms file. Type the forms filename, MYFORMS, and press ENTER . In reply to the question Create new form file (Y/N)?, type Y.

The screen displayed now is the edit menu (see Figure 1-4).

```
|                                                                                 |
|      EDIT    EDT FORM  EDT FLD    SWITCH     PREV PAG  NEXT PAG    FORM      MAIN|
|      FORM     ATTRIB    ATTRIB   LST FMT     FRM LIST  FRM LIST    UTILS     MENU|
|                                                                                 |
|                                                                                 |
___________________________________________________________________________________
```

Figure 1-4. The Edit Menu

From here you can select the functions for designing a new form, editing an existing form, or selecting the form utilities functions. These keys and their related functions are explained in chapter 2 and summarized in appendix B.

The cursor appears in the field beside Selected form:. When editing an existing form, you must enter the form name here. Leave this entry blank for now, and press f1 (EDIT FORM).

The top of the screen is clear except for the cursor. The keys used to design a new form appear at the bottom of the screen, as shown in Figure 1-5.

```
_____________________________________________________________________________________
|                                                                                   |
|     START      STOP     CENTER                                   REFRESH    SWITCH|
|     FIELD      FIELD     LINE                                                MENU |
|                                                                                   |
|                                                                                   |
_____________________________________________________________________________________
```

Figure 1-5. Form Design Keys

You can now begin to draw the form in Figure 1-6.

```
___________________________________________________________
|                                                         |
|     SOC. SEC. NO.                                       |
|     NAME                                                |
|     ADDRESS                                             |
|                                                         |
|     ZIP CODE                PHONE NO.                   |
|                                                         |
|                                                         |
___________________________________________________________
```

Figure 1-6. A Sample Form

Using the cursor control keys, position the cursor where the label SOC. SEC. NO. should begin.

Type the label SOC. SEC. NO. Create the field beside the label by pressing the f1 key (START FIELD) and by using the space bar to extend the field. When the field is the same length as that in Figure 1-6, press the f2 key (STOP FIELD). Follow this procedure with the other labels and their fields until your form resembles the sample form. If one field is created directly under another, as in Figure 1-6, the two fields may appear to be one. However, SIGEDITOR recognizes them as separate fields.

The f3 key (CENTER LINE) centers a line of the form. Position the cursor anywhere on the social security number line and press f3 . Your form looks like Figure 1-7.

```
       SOC. SEC. NO.

NAME

ADDRESS

ZIP CODE                      PHONE NO.
```

Figure 1-7. Sample Form with a Centered Line

Press ENTER . In the next screen, type the name of the new form and press ENTER to store the form and return to the main menu. At this point, you have created a forms file, designed a new form, and stored the form in the file.

Editing a Form

This section takes you through the steps of editing a form. The name of the newly created form is displayed beside Selected form: on your screen. Press f1 (EDIT FORM) to display your form on the screen. Press f8 (SWITCH MENU). A new set of keys for detailed form editing is displayed, as shown in Figure 1-8.

```
|                                                                                     |
|      ADD TO    SHORTEN   DELETE     HOLD          PLACE     CLEAR  REFRESH     SWITCH|
|      FIELD      FIELD    FIELD      FIELD         FIELD     HOLD                MENU |
|                                                                                     |
|                                                                                     |
```

Figure 1-8. Keys for Detailed Form Editing

To lengthen the NAME field on the form, position the cursor in that field (using the cursor control keys), and press the f1 key (ADD TO FIELD) several times. The field is lengthened one position each time you press f1 .

The f1 key can also create a new field in the form. Position the cursor under the field designated by the label ZIP CODE and type a new label, AGE. Then, press the f1 key three times to create a small field. Your form looks like Figure 1-9.

```
       SOC. SEC. NO.

NAME

ADDRESS

ADDRESS

ZIP CODE                     PHONE NO.

         AGE
```

Figure 1-9. Sample Form with a New Field Added

To shorten a field (remove spaces from the field) use the f2 key (SHORTEN FIELD). Position the cursor in the second address field and press the f2 key several times. One space is removed from the field each time you press f2 .

To delete an entire field, use the f3 key (DELETE FIELD). Position the cursor in the field with the label AGE and press the f3 key. The field no longer appears on the screen. Erase the label by spacing over the field. Your form now looks like Figure 1-10.

```
       SOC. SEC. NO.

NAME
```

```
        ADDRESS
        ADDRESS
        ZIP CODE                    PHONE NO.
```

Figure 1-10. Sample Form with a Field Deleted

The f4 key (HOLD FIELD) copies or moves a single field from one position to another.

Position the cursor on the field labeled PHONE NO. and press the f1 key. An H appears in the field to indicate that the field is being held. Next, move the cursor to the next line under the first position of the PHONE NO. field and press the f5 key (PLACE HOLD). A field identical to the held field appears beginning at the cursor. Label this field OFFICE PHONE. Your form looks like Figure 1-11.

```
               SOC. SEC. NO.
        NAME
        ADDRESS
        ADDRESS
        ZIP CODE                     PHONE NO.
                              OFFICE PHONE
```

Figure 1-11. Altered Sample Form

The hold field function can only be used with one field at a time. The H remains in a held field until you apply the function to another field or until you press f6 (CLEAR HOLD).

Press ENTER . The cursor appears under the name of your form. You have four options:

- Save the modified form under the original form name overwriting the old form.
- Save the modified form under a new form name preserving the old form.
- Press f7 (ABANDON EDITS) cancelling changes.
- Press f4 (RETURN TO EDIT) to make additional changes to the form.

Press f4 to return to the editing screen. Press f8 (SWITCH MENU) until the line drawing keys appear, as shown in Figure 1-12.

```
_____________________________________________________________________________________
|                                                                                   |
|       FROM       TO      HORIZON  VERTICAL      ERASE            REFRESH    SWITCH|
|     POSITION  POSITION   SINGLE    SINGLE                                    MENU |
|                                                                                   |
|                                                                                   |
_____________________________________________________________________________________
```

Figure 1-12. Line Drawing Keys

Three line types are available: single, bold, and double. The default is single. When the set of keys first appears, both the horizontal and vertical line types are set at the single type. To change the types press the f3 key (HORIZON) or the f4 key (VERTICAL).

To draw lines on a form, you must specify endpoints. For example, position the cursor on your screen to approximately the same position as the letter a appears in Figure 1-13. Press the f1 key (FROM POSITION).

```
______________________________________________________________________
|                                                                    |
|     e                                                                 f|
|       a                                                             b  |
|                                                                        |
|                                                                        |
|                      SOC. SEC. NO. ___________________                 |
|               NAME ________________________________________            |
|               ADDRESS _____________________________________            |
|               ADDRESS _____________________________________            |
|               ZIP CODE ___________ PHONE NO. ________                  |
|                                     OFFICE PHONE __________            |
|                                                                        |
|                                                                        |
|       d                                                             c  |
|     h                                                                 g|
__________________________________________________________________________
```

Figure 1-13. Location of Lines Drawn on Sample Form

Set the horizontal line type to bold by pressing the f3 key until the key on the screen reads 'HORIZON BOLD.'

Next, move the cursor with the cursor control keys to approximately the same position on the screen that the letter b appears in Figure 1-13. Press the f2 key (TO POSITION) to draw the bold line on the screen. Press the f1 key again, but this time set the vertical line type to double by pressing the f4 key until the key reads 'VERTICAL DOUBLE.' Move the cursor to the same position as the letter c and press the f2 key; a double line appears.

Repeat the process above, drawing a bold line across the bottom of the screen (from c to d) and a double line up the side (from d to a). Now set both line types to 'SINGLE' and draw another box around the form in the position of the letters e, f, g, and h in Figure 1-13.

SIGEDITOR automatically places appropriate connectors on the form. Chapter 2 describes how to specify connectors manually.

The f5 key (ERASE) initiates the erase function to erase part or all of a line within a form. Press the f5 key to turn erase on. Notice that the set of function keys at the bottom of the screen changes. These are the erase keys. To erase the lines on the side of the form, move the cursor to the upper right corner of the inside box on the screen and press f1 . Move the cursor to the lower right corner and press f2 . The line between the two endpoints just defined is erased. Repeat this process with the vertical line on the left side of the inside box.

Press ENTER . The name of the form is displayed in the next screen.

Press ENTER to save the changes and return to the main menu.

13 Chapter 2 Using SIGEDITOR

Introduction

This chapter gives a detailed description of SIGEDITOR's capabilities. They are:

- * Creating forms files.
- * Entering forms specifications.
- * Specifiying form field attributes.
- * Using additional editing features.
- * Using the form utilities.

Opening, Creating, and Listing Forms Files

To open a forms file, press f1 (SELECT FRMS FIL) from the main menu. Enter the form filename in the field provided and press ENTER . If the forms file does not exist, you can create a new forms file. To create a new forms file type "Y". You also have the option to enter a different forms filename, or to return to the main menu by pressing f8 . Form filenames follow MPE file naming conventions (up to eight alphanumeric characters are allowed, beginning with a letter). Once you specify a forms filename, the edit menu is displayed, and the names of the forms within the forms file are listed. The form names are listed in the default format of 16 rows, with eight form names per row. If there is more than one screenful of form names, you can list the next page of form names by pressing f6 . Use f5 to list the previous page. An alternate format lists one form name per row, along with detailed information regarding the size and date of creation and revision of each form. To switch to the alternate format, press f1 . To switch back to the default display format, press f1 again.

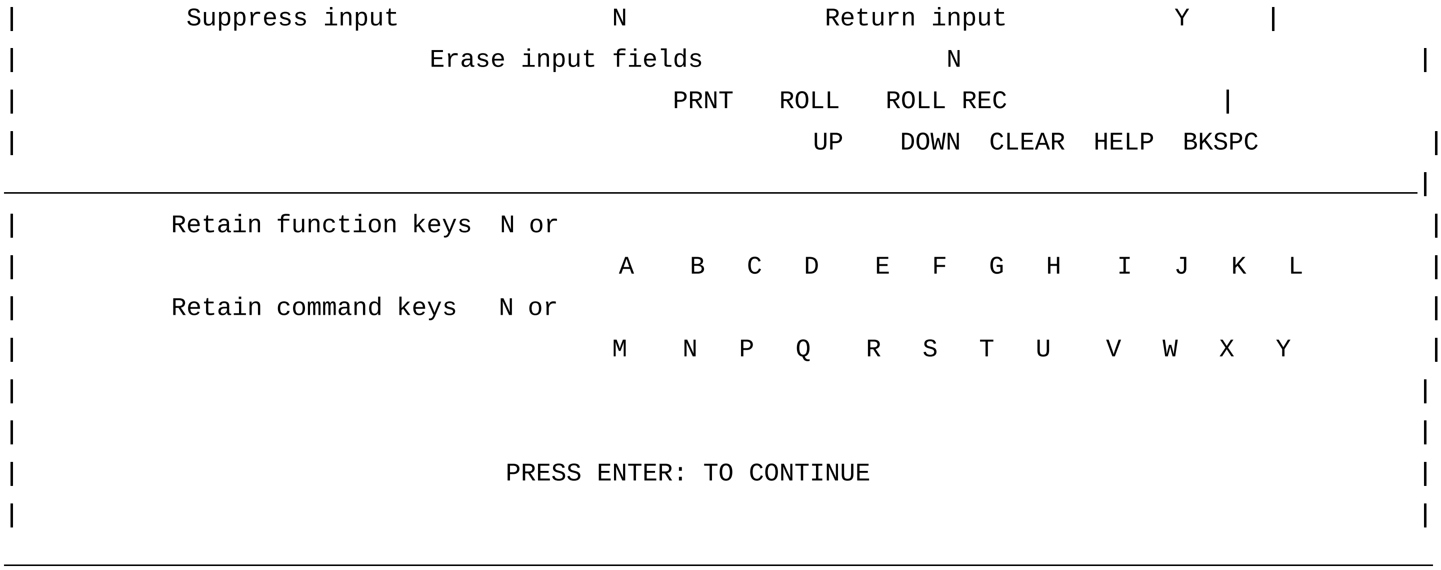
NOTE When you edit a forms file that was created with a version of SIGEDITOR prior to 8.00, the format is automatically converted. The new format is not compatible with RPG versions prior to 8.00. Be sure to back up form files before editing if you need to use an older version of RPG.

Editing Form Attributes

You can assign form attributes for each form and field attributes to each field in a form. Specifications or attributes describe how the form or field is formatted when used by a program. These settings are recorded and stored by SIGEDITOR. SIGEDITOR does not respond to the specifications, but stores them for later use by the RPG runtime library. See chapter 4 for more information about using forms with RPG. To edit form attributes, enter a valid form name in the edit main menu and press f2 (EDT FORM ATTRIB).

A set of form attributes appear on the screen after the main menu, as shown in Figure 2-1. The screen appears when you press the f2 key (EDT FORM ATTRIB). Figure 2-1 shows the default form attributes. These default values dictate how SIGEDITOR treats the form, unless you tab to a field and enter a new value.

```
______________________________________________________________________________|

|         Form Name                         FORM1     Start line number     01               |
|          Number of lines to clear    24             Uppercase only         Y      |
|          Sound alarm                 N             Override fields        N      |
|          Exception output            N             Forms cache *GLOBAL    N      |
```



```
|          Suppress input             N            Return input           Y     |
|                       Erase input fields               N                           |
|                                            PRNT   ROLL   ROLL REC               |
|                                                  UP    DOWN  CLEAR  HELP  BKSPC          |
____________________________________________________________________________________|
|         Retain function keys  N or                                                 |
|                                     A   B  C  D   E  F  G  H   I  J  K  L          |
|         Retain command keys   N or                                                 |
|                                     M   N  P  Q   R  S  T  U   V  W  X  Y          |
|                                                                                    |
|                                                                                    |
|                             PRESS ENTER: TO CONTINUE                               |
|                                                                                    |
_____________________________________________________________________________________
```

Figure 2-1. The Form Attribute Screen

After editing the form attributes, you must save or abandon the changes. To save them, press ENTER ; to abandon them, press f7 . In either case, you return to the edit main menu.

The next sections describe each form attribute.

Form Name

Legal values are letters A-Z, @, $, #, digits, or blanks.

The Form Name specification assigns a name to the form being designed. Within a forms file, each form name must be unique, but the same form name can be used in different forms files. A form name can have eight characters and must begin with a letter, @, $, or #; the remaining characters can be letters, digits, @, $, #, or embedded blanks. The only way to rename a form is to change the name from this menu.

Start Line Number

Legal values for this field are 1-24 and V.

The Start line number specification is the line number on which the form begins when it appears on the screen for data entry. The start line number is relative to the first line on the display (line one). Valid start line numbers are less than or equal to the number of lines on the screen (24 lines). If no value is entered, SIGEDITOR assumes that line one is the starting line. If you enter "V", the form has a variable starting line number obtained from a field in the RPG program as it is executing. The field is named in a "KSTART" continuation record in the workstation F-specifications.

NOTE Regardless of the start line number you set, the form appears to start on line one of the display. The form placement is not adjusted on the design screen, but is displayed correctly by an RPG program. For example, if you specify 10 as the start line number, you do not have to space down to the tenth line of the screen when creating the form. If you do, the form begins on line 20 when displayed by your program. Be careful not to design a form that exceeds 24 lines.

Number of Lines to Clear

Legal values for this field are 0-24.

Number of lines to clear specifies the number of lines to be cleared on the display before the form appears. Clearing begins at the line specified as the start line number. For example, if line three is given as the start line number and the number of lines to clear is eight, lines three through ten are cleared. The number of lines to clear plus the start line number must not exceed 24. If a value is not given, 24 lines (the default) are cleared.

NOTE The number of lines to clear and the start line number can be used to overlay one form with another. Use caution when overlaying a form with output fields with forms with input fields. All fields must be aligned or unpredictable results occur (see chapter 4 for more details). Clear all lines if you do not want to overlay forms.

Uppercase Only

Legal values for this field are Y and N.

If you specify Y (yes) in the Uppercase only specification, alphabetic characters entered by a user are returned to the screen and to your program in uppercase. If you specify N (no), lowercase data entered by the user are not returned to your program upshifted.

Sound Alarm

Legal values or this field are Y, N, and 01-99.

If you specify Y (yes) in the Sound alarm specification, an audible alarm sounds when the form is displayed. If you specify N (no), no alarm sounds. If you specify an indicator 01 through 99 and turn it on in your program, the alarm sounds when the form is displayed.

Override Fields

Legal values for this field are Y, N, and 01-99.

An Override fields operation allows you to override (alter) fields in a form when the form is redisplayed. Only those fields modified as a result of an override operation are redisplayed. You can use an override operation if the user enters invalid data into a field and you want to flag the field in error.

During an override operation, if an indicator 01 through 99 is specified and turned on, then:

- If a field has an indicator in the output data entry (see "Field Attributes") and the indicator is off, the data entered in that field is unchanged. Any field with a Y or N in the output data entry is also unchanged.
- If a field has an indicator in the output data entry and the indicator is on, the field is displayed with data from the program.If the field is also input capable, data that was entered into the field by the user is lost.
- Form and field attributes that are controlled with indicators will be changed according to the current state of those indicators.

If you specify Y (yes) in the Override fields specification, the override operation is performed each time the form is displayed.

NOTE Using Y is not recommended because only those fields with indicators on will be displayed when the form is initially displayed. Screen resident constants are not indicator controlled, thus they will never be displayed.

If you specify N (no), no override operation is performed on that form.

Exception Output

Legal values for this field are Y and N.

Exception output is used to force the immediate display of a form. If Y (yes) is specified, the form is output at exception time. If N (no) is specified, the form is output prior to the next read from the WORKSTN file.

Forms Cache (*GLOBAL)

Legal values for this field are Y and N.

The Forms cache (*GLOBAL) determines the form name stored in the for ms cache directory when the form is downloaded.

NOTE Forms can be downloaded using the DOWNLOAD command of the Process Monitor software product, PROCMON. Currently, only the 2394A and 700/94 terminals support forms downloading. For more information, see the *PROCMON/3000 Reference Manual*.

If Y (yes) is specified, the name of the form in the forms cache directory consists of the form name and *GLOBAL. This allows a form that is in multiple forms files to be accessed as the same form from the forms cache. If N (no) is specified, the name of the form in the forms cache directory consists of the form name and the form filename that contains the form.

Suppress Input

Legal values for this field are Y, N, and 01-99.

If Y (yes) is specified in the Suppress input specification, or an indicator is specified and the indicator is on, all input fields on the form will be returned to the RPG program as blanks. If N (no) is specified or an indicator is specified and the indicator is off, the input field will be returned to the RPG program normally.

NOTE If you specify Y (yes), input can never be read from the form.

Return Input

Legal values for this field are Y and N.

If Y (yes) is specified in the Return input specification, all input fields in the form are returned to your program, even if the user did not enter any data in the form. If N (no) is specified, data from input fields is returned only if the user enters data. If the user does not enter any data on the screen, all input fields are filled with blanks. Information does not have to be entered into mandatory enter fields (see "Field Attributes").

Erase Input Fields

The legal values for this field are Y, N, and 01-99.

The Erase input fields specification erases the input fields on a form. It is helpful to erase input fields when a user must enter information into the same form so that the form does not have to be redisplayed. If Y (yes) is specified in the Erase input fields specification, input fields are erased each time the form is displayed. If N (no) is specified or this entry is left blank, no erase input fields operation is allowed.

If an indicator is specified in this field and is enabled in your program, the erase input fields operation is performed on the form only when the indicator is turned on. If the indicator is not on, the form is displayed normally.

Retain Function Keys

Legal values for this field are Y and N.

Function keys are used to return status information to the RPG program. When the user presses a function key, a four-digit numeric status code is placed in element one of the status array. You can test function keys f1 through f8 by examining element one of the status array for the codes shown in Table 1.

If Y (yes) is specified in the Retain Function Keys specification, the function keys that were enabled for the previous form are retained when this form is displayed.

If N (no) is specified, the function keys active for the previous screen are not retained, and the user can enable new function keys for this form. Table 1 describes each function key.

Table 5: Function Keys

Function Key	Description	Status Code
f2	print screen	1121
f3	roll up	1122
f4	roll down	1123
f5	clear	1124
f6	help	1125
f7	record backspace	1126

Enable a function key by placing any character in the space identified by the function number. The function key is enabled for a specific use that you determine in the RPG program. If a space is left blank, that function is not enabled.

NOTE The f1 key is reserved for the "command" key. The f8 key is used as the "dup" (duplicate) key.

Print Screen Key

Legal values for this field are Y and N.

If you enable the Print key for the current form, you determine its function. If you do not enable the Print key for the current form, pressing the Print key causes the contents of the display screen to be printed. The default device is LP, using the filename RSIPRINT. An asterisk (*) to the left of the Print key label indicates that the Print key is not enabled for this form.

Retain Command Keys

Legal values for this field are Y and N.

Command keys are used in RPG programs to set indicators (KA-KN, KP-KY). When the user presses a command key, indicators are set that the program can use to perform the desired function. The command keys are independent of the function keys.

If Y (yes) is specified in the Retain command key specification, the command keys that were active for the previous form are retained when this form is displayed.

If N (no) is specified, the active command keys are not kept and you can enable new command keys for this form. Letters A through N on the Form Specification Screen correspond to command key indicators KA through KN. Letters P through Y correspond to indicators KP through KY. Command keys are enabled for a form by placing any character in the field corresponding to the command key. If a field is left blank, the corresponding command key is not enabled. The command key indicators are used in the RPG program to control the flow of operation. Refer to the HP RPG/XL Reference Manual, chapter 11 for a detailed list of command keys.

Editing Field Attributes

To edit field attributes, enter the form name in the field provided on the edit main menu and press f3 (EDT FLD ATTRIB.). The field attribute keys shown in Figure 2-2 are displayed.

```
|                                                                                 |
|       NEXT      PREV     FIRST      SET                 DEFAULT   SEE/SET    EDIT|
|       FIELD     FIELD    FIELD     FIELD                ATTRIB.   ATTRIB.    MENU|
|                                                                                 |
|                                                                                 |
```

Figure 2-2. Field Attribute Keys

The first three keys allow you to move the cursor from field to field. Pressing f1 (NEXT FIELD) moves the cursor to the next field on the form according to line and column numbers. The next field is found on the basis of which field begins in the next highest column or row, not on the basis of the field's assigned sequence numbers. When the cursor is positioned in the last field and the f1 key is pressed, the cursor moves to the first field. In contrast, when the cursor is positioned in the first field and the f2 (PREV. FIELD) key is pressed, the cursor moves to the last field. The f3 key (FIRST FIELD) moves the cursor to the first field on the form.

The form is also displayed on the screen, as shown in Figure 2-3.

```
|                                                 |
|     NAME A#########################             |
|     ADDRESS  A###################################|
|     PHONE NO.  A#########                       |
|     SOC. SEC. NO.  A#############               |
```

Figure 2-3. Sample Form

The letter in the first position of the field indicates the data type of that field. The possible data types are summarized in Table 2-2. The number signs (#) show how far the field extends

Table 6: Allowable Field Data Types

Letter	Data Type
A	alphabetic
B	alphanumeric
N	numeric
S	signed
D	decimal

From this screen, you can change the attributes of individual fields. Once the cursor is positioned in the field whose attributes are to be changed, pressing f7 (SEE ATTRIB.) displays the current attributes of that field, as shown in Figure 2-4.

```
|                                                                                     |
|                   RPG Screen Interface Attribute Specification                      |
|                                                                                        |
|                Form File:  formfile.group.account            Form:  formname           |
|                                                                                        |
|
----------------------------------------------------------------------------------|
|                                                                                        |
|     Field Name:       Sequence Number:                                                 |
|     Position Cursor: N    Protect Fields: N    High Intense: N    Blink  Field: N       |
|     Security Video:  N    Inverse Video: Y     Underline:     N    Output  Field: N     |
|     Constant Type (C=constant, M=Message, " " = User Program)      Input  Field:  N     |
|        Message Set No:         Message No:                                             |
|     Data Type  (A+Alpha, B=Alphanumeric, D=Decimal, N=Numeric, S=Signed)       B       |
|                                                                                        |
|                                                                                        |
|     Mandatory Enter   N          Mandatory Fill   N        Enable Duplication   N      |
|                                                                                        |
|                                                                                        |
|     Adjust Fill (Z = Zero, B = Blank):           Self Check (T = 10, E = 11)           |
|                                                                                        |
|                                                                                        |
```

Figure 2-4. The Field Attribute Screen

As with the form specifications, default values are assigned to the field attributes when a field is created. When the field attributes are displayed, the tab key moves the cursor from one attribute to the next. To change an attribute position, use the tab key to position the cursor at the attribute and type the new value.

When you have set the desired attributes, press f4 to save the new values. To return to the default attributes, press f6 . When you are done, press f8 to return to the edit menu. Each field attribute is described below.

Field Name

The Field Name entry contains the name of the field. A field name can be up to six characters, beginning with a letter or @, $, or #. The remaining characters can be letters, digits, or @, $, or #.

NOTE If you leave the field name blank, SIGEDITOR creates a name for the field. This name consists of the letters FLD followed by a number.

Sequence Number

The Sequence Number establishes a relationship between the sequence in which fields appear on the form and the sequence in which the fields are placed in the RPG program input/output (I/O) buffer. When you create a form, fields are sequenced according to line and column number. When the form is edited after the initial design, sequence numbers are assigned to fields according to the order in which they are created, not according to line and column numbers.

You can change the order of the fields' placement in the buffer by changing the sequence number of each field. However, when changing the sequence of a form's fields, be aware that previous RPG input and output specifications for that form no longer apply. I/O specifications must reflect the sequence in which fields are placed in the buffer. If you change the sequence numbers of a form's fields without generating new I/O specifications, the data is not stored in the correct buffer locations.

NOTE If you change the sequence numbers of a form's fields, regenerate I/O specifications and recompile the program. (See "Generating I/O Specifications.")

Position Cursor

Legal values for this field are Y, N, and 01-99.

The Position Cursor attribute positions the cursor at a specific input field when a form is first displayed.

If Y (yes) is specified, the cursor appears at the first position in this field, unless another field in the same form has an indicator specified for Position Cursor and the indicator is on. If so, the cursor appears at the first position of the first field whose indicator is on. Specify Y for only one field in a form. A Y in any following field has no effect.

If N (no) is specified, the cursor will not appear in this field unless it is the first unprotected field in the form, no other field uses a Y in the Position Cursor field, and no other field has an indicator that is specified for Position Cursor. If more than one field in a form has indicators specified in this entry, and the indicators are on when the form is displayed, the cursor appears in the first field whose indicator is on.

Protect Field

Legal values for this field are Y, N, and 01-99.

If Y is specified, or the field uses an indicator and the indicator is on, the user cannot enter data in the input field (the field is protected). If N is specified, or a field indicator is used and it is off, the user can enter data.

High Intense

Legal values for this field are Y, N, and 01-99.

If Y (yes) is specified, the field is displayed with high intensity. If N (no) is specified, the field is displayed with normal intensity. If an indicator is specified, the field is displayed with high intensity when the indicator is on.

Blink Field

Legal values for this field are Y, N, and 01-99.

If Y (yes) is specified, the field blinks; if N (no) is specified, the field does not blink. If an indicator is specified, the field blinks when the indicator is on.

Security Video

Legal values for this field are Y, N, and 01-99.

If Y (yes) is specified, the field is a nondisplay field. Information stored in the field is not echoed to the screen when the form is displayed or when the user enters information into the field. If N (no) is specified, the information in the field is displayed. If an indicator is specified, the field is a nondisplay field when the indicator is on.

NOTE If security video is specified with high intensity, inverse video, or underline, these attributes are ignored. On a terminal that does not support security video, RSI selects the alternate character set for unprotected input fields.

Inverse Video

Legal values for this field are Y, N, and 01-99.

If Y (yes) is specified, the characters in the field appear in inverse video. If N (no) is specified, the characters appear in normal video. If an indicator is specified, the field appears in inverse video when the indicator is on.

Underline

Legal values for this field are Y, N, and 01-99.

If Y (yes) is specified, the field is underlined. If N (no) is specified, the field is not underlined. If an indicator is specified, the field appears underlined when the indicator is on.

Output Field

Legal values for this field are Y, N, and 01-99.

If Y (yes) is specified, the field is an output field. An output field is a field in which variable data, constant data, or a message is written to the screen when a form is displayed. If the field is not an input or input/ output field, data in that field cannot be changed by a user. An output field is used as follows:

- If C is specified in the constant type entry, the field displays the literal specified for this field.
- If M is specified in the constant type entry, and the message set number and message number entries are specified, these numbers are used to obtain a message from the merged catalog file, which is then displayed in this field at runtime. If the message number entries are not specified, the message set number and message number entries are taken from the RPG output buffer at runtime and used to obtain the message text from the catalog file.
- If the constant type entry is left blank, the field is used to display program data from the output buffer.

If N (no) is specified, the field is not an output field and the constant type field is ignored. If an indicator is specified and the constant type entry is M or C, the results are as shown in Table 3. If an indicator is specified and it is on, data from the program is displayed. Otherwise, blanks are displayed.

Table 7: Output Field and Indicators

	Indicator is turned ON	Indicator is turned OFF
M is the constant type.	The message identified by the message set number and message number from your program is displayed.	The message identified by the message set number and the message identifier in the field is displayed. If the field is blank, blanks are displayed.
C is the constant type.	The data from your program output record area is displayed.	The constant data from the forms file is displayed.
Override operation is performed.	The data given by your program or the message identified by your program is displayed.	The field is unchanged.

Input Field

Legal values for this field are Y and N.

If Y (yes) is specified, the field is an input field and may be defined in an input specification in your program. If the field is defined as both an input and an output field, and initially contains a constant, the constant may be overwritten prior to entry of the data. However, the output field must have an indicator on for this to happen. If N (no) is specified, the field is not an input field and cannot accept data entered by a user.

Constant Type

Legal values for this field are C or blank.

If C is specified in the Constant type specification, constant information given in the field is displayed in the output field. If the field is blank, blanks are displayed.

If a blank is specified, information from the program output buffer area is displayed.

If M is specified in the constant type entry, and the message set number and message number entries are specified, these numbers are used to obtain a message from the merged catalog file, which is then displayed in this field at runtime. If the message number entries are not specified, the message set number and message number entries are taken from the RPG output buffer at runtime and used to obtain the message text from the catalog file.

If the message text cannot be obtained using the message I.D. provided, blanks are displayed.

Message Set Number

Legal values for this field are M or blank.

If the constant type specified is M (message), you can specify a message set number representing the message set number within a catalog file that is to be merged (see "Referencing Messages and Merging Catalog Files"). The message will not be merged until you use the merge catalog option in either the main menu or the form utilities menu.

Message Number

Legal values for this field are M or blank.

If the constant type specified is M (message), you can specify a message number representing the message number within a message set of a message catalog file.

Data Types

Legal values for this field are A, B, D, N, S, and blank.

The letter given in this entry indicates the type of data that a user can enter in an input field. Data type is ignored for a field that is output only. The data types are described in Table 2-4.

Table 8: Data Types

Data Type	Description
A	The field can only contain alphabetic data.
B or blank	The field can contain any alphanumeric data.
N	The field can only contain numeric data, commas, periods, plus, or minus signs. The editing characters are stripped when the data is moved to the program buffer, except for the sign, if any.
S	The field can contain only signed numeric data with one position of the field reserved for a sign. The field can be from 2 to 16 characters long. Although the sign for field is displayed on the screen, the buffer space allocated for the field is one character less because the sign is overpunched on the low-order digit.
D	The field can only contain numeric data, commas, periods, plus, or minus signs. The edit characters (comma, period, plus, and minus) are not stripped from this field before it is sent to the user program. This field can be defined as alphanumeric in the RPG program.

Examples of numeric and signed numeric data types are shown below.

Entered on screen In buffer

```
Numeric:                1234.5             12345
Signed numeric:           -1234              123M
                             or
                          1234-              123M
  A blank is represented by a   .
```

Mandatory Enter

Legal values for this field are Y and N.

If Y (yes) is specified, the user must enter at least one character in the field before input from the form can be returned to your program. If N (no) is specified, information does not need to be entered in that field.

Mandatory Fill

Legal values for this field are Y and N.

If Y (yes) is specified and the user enters at least one character in the field, the user must enter nonblank characters in all positions in the field. Nonblank characters are any characters entered from the keyboard, except the blank space. Alternatively, the field can be left completely blank. If Y is specified for Mandatory Fill for a signed numeric field, the user must enter a trailing minus sign for the negative fields. If N (no) is specified, the field does not have to be completely filled to be accepted.

Enable Duplication

Legal values for this field are Y and N.

If Y (yes) is specified, the duplicate (Dup) key can be pressed in the field. When the the user presses the Dup key, f8 , the cursor advances to the next field. The position of the cursor and the remainder of the field are filled with the duplicate character value (hexadecimal 1C), which is displayed as a single character. The Dup key does not do actual duplication. The user program must check for the duplicate character and do the duplication. If N (no) is specified, the "dup" key is not valid and causes an error.

NOTE This feature is generally used in data entry programs to simulate the dedicated "Dup" key found on specialized data entry equipment.

Adjust Fill

Legal values for this field are Z and B.

If Z is specified (indicating right-justified, zero-filled), information entered in the field is right-justified with all unused positions in the field filled with zeros when the field is sent back to your RPG program. If B is specified (indicating right-justified, blank-filled), the information entered in the field is right-justified with all unused positions filled with blanks when the field is sent to your RPG program. If this entry is left blank, right-justified, blank-filled is assumed on input for signed numeric fields. No-adjust, no-fill is assumed for other fields.

NOTE Adjust/fill has no effect if Mandatory Fill is specified for the field.

Self-Check

Legal values for this field are found in appendix C.

The self-check entry is used only for numeric fields. It is used, for example, to catch errors on entering items such as account numbers. The number is initially created by a user program, using an appropriate algorithm, to compute a check digit, which is then appended to the right-most end of the number. For example, a 6-digit number plus the check digit would become a 7-digit account number. When the number is entered on the screen and self-check is specified, the number is checked, again using an appropriate algorithm, to ensure the number was entered correctly. Hewlett-Packard provides two methods for checking the check digit.

If a T is specified in the self-check field, a modulus 10 self-check test is performed. If an E is specified, a modulus 11 check is used. See appendix C for detailed information on these algorithms. If the entry is left blank, no check is made. You must use the same algorithm to generate the check digit as that used to check it in your program. The number may contain leading blanks, embedded blanks, and punctuation. The checking algorithm treats these characters as zeros.

Setting the Attributes

When you are finished making changes to the attributes of a field, you can either save or cancel them. To save them, press f1 (SET FIELD). To cancel them, f7 (SEE ATTRIB.). Both f1 and f7 will return you to the form.

If the field is an input field, the data type appears as the first byte in the field when the form is redisplayed. Any video attributes that were specified are also displayed.

When the field attributes have been modified, the pound signs (#) shown in Figure 2-3 are changed to asterisks (*). This indicates that the attributes of a field have been modified. When you have set all fields in the form with the correct attributes, press f8 (EDIT MENU) to return to the edit menu.

Additional Editing Features

This section describes additional SIGEDITOR editing capabilities that were not described in chapter 1.

Creating Fields with the Tab and Right Arrow Key

As described in chapter 1, the space bar creates a field when in create form mode. Once the f1 key (START FIELD) designates the starting point of a field, pressing the space bar extends the field to the desired end point.

There are two other ways to extend the field. First, you can use the right arrow key. However, unlike the space bar, the right arrow key does not show the field in inverse video until you press the space bar or the f2 key (STOP FIELD). When using the right arrow key to extend a field, you must remember to stop the field.

The second method of extending a field is to use the tab key. The tab key extends the field by ten spaces at a time, relative to the starting point of the field. For example, if you press the f1 key (START FIELD), followed immediately by the tab key, the field is extended ten spaces. If, however, you press the space bar first to extend the field four spaces, pressing the tab key extends the field by six more spaces, or ten spaces from the start point. The tab key actually extends the field eleven spaces, but the last space is erased when the (STOP FIELD) key is pressed. When you use the space bar, the cursor leads the field as the field is extended. In contrast, the cursor is included in the field when you use the tab key.

Stop Field Key

The f2 key (STOP FIELD) appears in the set of keys used to create a form. As mentioned earlier, you must use the f2 key to indicate the end of a field. If you do not use this key to end a field, and proceed to another line to create a new field, SIGEDITOR assumes that the field extends to the end of the line. Although the whole field might not appear on the screen in reverse video, it is recorded by the program as extending to the end of the line. Delete and recreate the field if you are not sure that you correctly stopped the field.

Other Keyboard Edit Keys

Although it is not recommended, you can modify a form with keys not included in the function keys f1 through f8 . The additional six keys are shown in Figure 2-6. These keys can be useful when editing fields; however, it is good practice to use the eight function keys established by SIGEDITOR.

```
INS       DEL
LINE      LINE
INS       DEL       CLEAR     CLEAR
CHAR      CHAR      LINE      DSPLY
```

Figure 2-5. Additional Edit Keys

These edit keys perform the same functions in SIGEDITOR that they normally perform, but they do not act exactly the same as the eight function keys.

For example, the f2 key (SHORTEN FIELD) removes a space from the field containing the cursor and leaves the rest of the line intact. SIGEDITOR immediately updates the field table to reflect any changes. The delete character (DEL CHAR) key works differently. It deletes a character from the field in which the cursor appears, but it also moves the rest of the line one space to the left. The changes to the field table are not immediate and are not reflected until the form is saved in memory.

The same differences apply to the f1 key (ADD (TO) FIELD) and the insert character key. The ADD (TO) FIELD key adds a space to the field; the insert character key inserts a space to the whole line. If you are not careful, fields at the end of the line can be truncated.

Form Utilities

Enter the utilities menu, shown in Figure 2-11, by pressing f7 (FORM UTILS) from the main menu.

```
______________________________________________________________________________________
|                                                                                    |
|      HELP                DELETE     COPY       MERGE     PRINT   GEN I/O  EXIT FRM|
|     AREAS                 FORM      FORM      CATALOG    SPECS    SPECS     UTILS |
|                                                                                    |
______________________________________________________________________________________
```

Figure 2-6. Utilities Menu

Each of these keys is described in the next sections.

Printing Specifications

In SIGEDITOR you can print a copy of the form and field specifications in two places; in the form utilities menu, using the f6 (PRINT SPECS) key, and in the main menu, using the f5 (PRINT SPECS) key. The resulting printout includes a visual representation of the form, the form name, and the global form attributes. For each field in the form, the name, position in the form, and the attributes are printed. In general, SIGEDITOR only prints what is necessary to understand the set up of the form and fields.

The formal file designator is SIGLIST.

Generating I/O Specifications

You can generate RPG input/output specifications either using the f7 key (GEN I/O SPECS) on the form utilities menu, or using the f6 key on the main menu. You must enter the name of a valid forms file and output file before generating the I/O specifications. Press ENTER to begin. When the procedure is complete, press f8 to return to the previous menu.

If the output file already exists, SIGEDITOR sends a verification message to the screen asking if the previous contents of the file can be purged. If the answer is no, SIGEDITOR requests another filename. Otherwise, a new message, GENERATING INPUT/OUTPUT SPECIFICATIONS is sent to the screen. You are then returned to the utilities menu. You can merge the file containing the I/O specifications with the RPG program to avoid entering information that is already available (see the $COPY option description in the *HP RPG/XL Reference Manual*). If a field's size, position, name, or sequence number is changed, the changes must be reflected in the program by regenerating the I/O specifications and copying the new specifications into the program.

NOTE When generating I/O specifications, SIGEDITOR assigns a field name to any field you leave unnamed. This default field consists of the letters FLD followed by the field's sequence number.

Deleting a Form

Use the delete form option to delete a form. Press f3 (DELETE FORM) on the form utilities menu. The delete form screen is displayed (see Figure 2-9).

```
_______________________________________________________________
|                                                             |
|              Forms file:                                    |
|                                                             |
|              Form name:                                     |
|                                                             |
|                                                             |
|                                                             |
_______________________________________________________________
```

Figure 2-7. Delete Form Screen

Here you must supply the name of the form to be deleted, as well as the forms file containing that form. The forms filename defaults to the name of the currently opened form file. Once the correct information is entered, press ENTER to delete the form. Press f8 to return to the form utilities menu.

Copying a Form

The copy form option copies a form from one forms file to another. When the f1 key (COPY FORM) is pressed, the Copy Form screen is displayed (see Figure 2-8).

```
____________________________________________________________________
|                                                                  |
|                   Forms file:                                    |
|    FROM:                                                         |
|                   Form name:                                     |
|                                                                  |
|                                                                  |
|                                                                  |
|                   Forms file:                                   |
|    TO:                                                           |
|                   Form name:                                     |
|                                                                  |
|
```

Figure 2-8. Copy Form Screen

The space allowed to enter the name of the FROM forms file is long enough to specify the fully qualified filename of a forms file. Once the designated forms file and form names are given and the ENTER key is pressed, SIGEDITOR simply copies the form to the specified forms file under the new form name.

Press f8 to return to the form utilities menu.

Referencing Messages and Merging Catalog Files

In SIGEDITOR, a message is informational text that may be displayed at runtime. There are two ways to reference messages when a program is run. The first method is to specify a message set number and message number in the message set number and message number fields when setting field attributes for a given field.

The message set number and message number specified in the field must correspond to a message set number and message number in a catalog (or message) file. The *RPG Reference Manual* describes how to create a catalog file. The catalog file contains the actual message; the field attribute contains the set number and message number. The second method of referencing a message is to identify the message in the RPG program. If this is done, the set number and message number on the field attributes screen can be left blank if desired. See the description of the output field attribute in this chapter for more details. The catalog file is set up in exactly the same way, but the message is referenced from the program, not from the form. Examples of both methods are given in chapter 4. The two methods can be used together to provide a default message (from the field attributes screen) if a set number and message number are not provided by the RPG program. A merge operation retrieves a message that is referenced by a message set number and message number and stores the message as part of the form. Merging can be performed any time after a forms file is created using the f5 key on the utilities menu, or by using f1 on the main menu. If you do not merge a forms file with a catalog file, messages must be retrieved at runtime. However, merging prior to runtime speeds the running of the program. Therefore, saving the merge until runtime is less efficient.

NOTE If you merge catalog messages with forms files, any subsequent modifications to the catalog messages require that the catalog be remerged in order for these changes to show in the program at runtime. The messages must also be remerged after editing a form that uses merged messages.

When merging a catalog file, you must enter both a valid catalog and forms file. Press ENTER to merge the messages. To exit the merge utility, press f8 .

When the merge operation is performed, only the messages identified by a set number and message number on the field attributes screen will be stored as part of the form. Messages that are identified with a set number and message number from the RPG program cannot be merged using the SIGEDITOR.

14 Chapter 3 Application Help Facility

Introduction to the Application Help Facility

The application help facility is part of the RPG Screen Interface (RSI). At runtime, RSI uses "help areas" and "help forms" created by SIGEDITOR. For applications converted from the IBM S/36, the SIGEDITOR converts the help (or H) specifications to the RSI format as part of the convert S and D specifications function.

Help Areas

Help areas are imaginary areas that are overlayed on the application form. The help area can be as small as a single character or as large as the entire screen. Up to 256 help areas can be associated with an individual form.

When an application form is displayed and the user presses the f6 (HELP) key, RSI notes the cursor position at the time f6 is pressed and scans a list of help area definitions for the first one that covers the cursor position. The help area definition contains the name of the first help form to display. Additional help forms, if available, can be displayed using the f3 (ROLL UP) and f1 (ROLL DOWN) keys. The user must press the ENTER key to return to the application form.

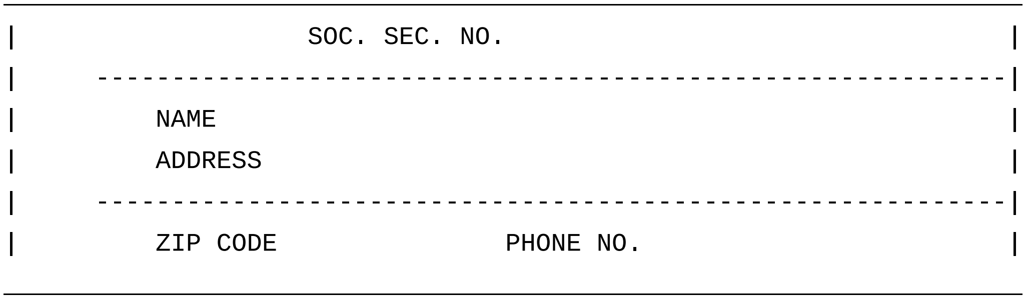
Help Forms

Help forms are ordinary RSI forms, except that they are made up of text only. They can start on any line (01 through 24) and clear up to 24 lines. Help forms can be overlaid on other help forms, but not the application form. Help forms cannot contain fields, video enhancements, or message constants. They cannot use the variable starting line number feature. If a help form contains unsupported specifications, they are ignored when the form is displayed.

To use the RSI application help facility in an RPG program, you must first prepare the program with DS capability. The application form must have the f6 (HELP) key disabled. No additional RPG programming is required to use the basic RSI application help facility. If you want to use the suppress selection or boundary features, you must enter RPG calculation specifications to control these indicators.

Defining Help Areas

From the main menu press f7 (FORM UTILS), this takes you to the form utilities menu. From the form utilities menu press f1 (HELP AREAS). Then, to define a help area, move the cursor to where you want the upper left corner of the help area and press the f1 (UPPER LEFT) key. Next, move the cursor to where you want the lower right corner of the help area and press the f2 (LOWER RIGHT) key. At this point, the help area is outlined on the screen as shown in figure 2-6.



```
|                    SOC. SEC. NO.                        |
|     ----------------------------------------------------|
|         NAME                                            |
|         ADDRESS                                         |
|     ----------------------------------------------------|
|         ZIP CODE              PHONE NO.                 |
```

Figure 3-1. Defined Help Area

If you are not satisfied with the help area, you can start over by pressing the f5 (REFRESH FORM) key. The current help area is the only one discarded. To see all help areas defined for the form, press the f6 (SHOW ALL AREAS) key.

The last step in defining a help area is to assign help area attributes. The attributes are assigned by pressing f3 (HELP ATTR) and then entering the help attributes in the spaces provided on the help attributes form. The attributes are defined in the next section. If you are not satisfied with the help area, you can start over by pressing the f7 (ABANDON) key. The help area definition is not retained until the help attributes are set by pressing the ENTER key. Once you press the ENTER key, you are returned to the application form.

Setting, Changing, and Using Help Attributes

Each help area has a set of attributes associated with it. These are:

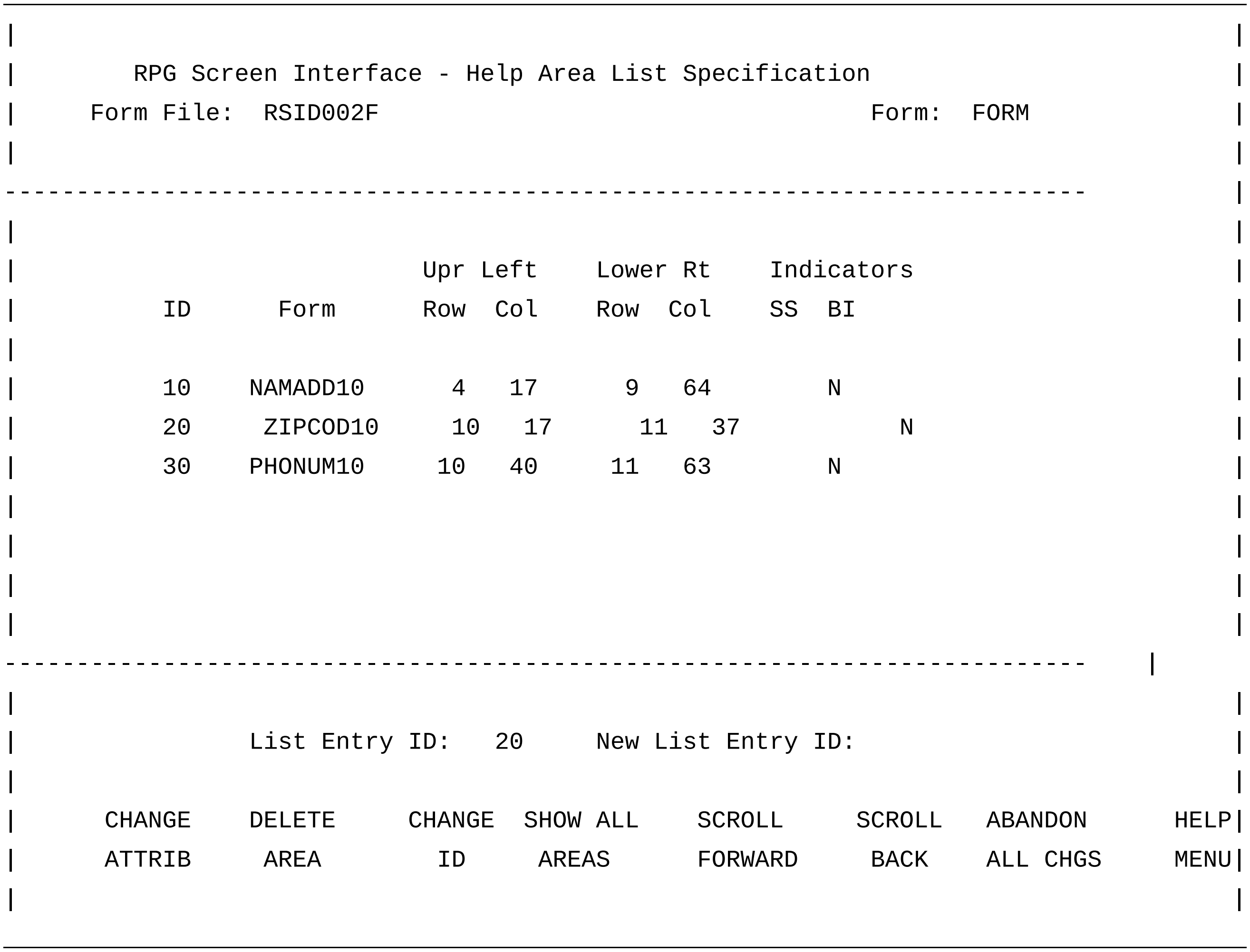
- * The name of the help form to be displayed.
- * The row and column numbers of the help area's corners.
- * Information that tells RSI how to handle the help area at runtime.

You set help area attributes during the help area creation process. The help area attribute specification screen is illustrated in figure 3-2.

```
|                                                                       |
|            RPG Screen Interface - Help Area Attribute Specification   |
|          Form File:  RSID002F                             Form:  FORM |
|                                                                       |
----------------------------------------------------------------------------|
|                                                                       |
|          Help Form Name:                                              |
|                                                                       |
|          Upper Left Row:                    Upper Left Column:        |
|                                                                       |
|          Lower Right Row:                   Lower Right Column:       |
|                                                                       |
|                        Suppress Selection Indicator:                  |
|                                                                       |
|                             Boundary Indicator:                       |
|                                                                       |
----------------------------------------------------------------------------|
|                                                                       |
|                                                              ABANDON  |
|                                                                       |
```

Figure 3-2. Help Area Attribute Specification Screen

You can change the attributes using the change attribute operation. The help area list specification screen used in the change attribute operation is illustrated in figure 3-3.



```
|                                                                       |
|       RPG Screen Interface - Help Area List Specification             |
|    Form File:  RSID002F                          Form:  FORM          |
|                                                                       |
---------------------------------------------------------------------   |
|                                                                       |
|                      Upr Left    Lower Rt    Indicators               |
|        ID      Form      Row  Col    Row  Col    SS  BI               |
|                                                                       |
|        10    NAMADD10      4   17       9   64        N               |
|        20     ZIPCOD10     10   17      11   37            N          |
|        30    PHONUM10     10   40     11   63        N               |
|                                                                       |
|                                                                       |
|                                                                       |
|                                                                       |
---------------------------------------------------------------------   |
|                                                                       |
|             List Entry ID:   20     New List Entry ID:                |
|                                                                       |
|     CHANGE    DELETE     CHANGE  SHOW ALL    SCROLL     SCROLL   ABANDON      HELP|
|     ATTRIB     AREA        ID     AREAS      FORWARD     BACK    ALL CHGS     MENU|
|                                                                       |
```

Figure 3-3. Help Area List Specification Screen

The following sections explain each help area attribute in detail.

List Entry ID

The list entry ID is a number that identifies a specific help area during the help area update/delete process. The numbers are regenerated each time help area update/delete is initiated.

Help Form Name

The help form name is the name of the first appplication form containing help text to be displayed when this help area is selected. The help form name must be eight characters long, begin with an alpha character, and end with two digits from 00 to 99.

Help Form Families

Help forms can be grouped into "families." A family of help forms share the same first six characters of their names, but the last two numeric characters vary (for example, HELPFM01, HELPFM02, HELPFM03, etc.). Help form families are used during roll key processing, which is described later in this chapter.

Upper Row

This is the upper row of the help area border. The upper row cannot be less than the start line of the application form. It must be in the range of 01 to 24. The only exception is in the case of the "NULL HELP AREA" (see roll key processing). When a "NULL HELP AREA" is created all row and column attributes must be zero (0).

Left Column

This is the left column of the help area border. The left column must be in the range of 01 to 80, except in the case of a "NULL HELP AREA" when it must be zero (0).

NOTE Where the upper row number and the left column number meet is where the upper left corner of the help area is defined.

Lower Row

This is the lower row of the help area border. The lower row must be in the range of 01 to 24, except in the case of a NULL HELP AREA when it must be zero (0). The lower right row number cannot be less than the upper left row number.

Right Column

This is the right column in the help area border. The number of the right column must be in the range from 01 to 80, except in the case of a null help area when the boundary must be zero (0). The lower right column number cannot be less than the upper left column number.

NOTE Where the lower row number and the right column number meet is where the lower right corner of the help area border is defined.

Suppress Selection Indicator

The suppress selection indicator hides help areas at runtime. It must be blank or in the range 01 to 99. The default is blank. When an application form that contains help areas is displayed, only the help area definitions in which the suppress selection indicator is blank or set off (in the RPG program) are added to the global help area list. Additionally, help areas in the global help area list are deleted from the list if the suppress selection indicator is set on when a new application form is displayed. See "Runtime Global Help Area List Management" for more discussion of the global help area list.

Boundary Indicator

The boundary indicator creates borders in the help area list, in addition to the first and last entries. These boundaries are used in roll key processing, which is described below. Using one or more boundary indicators divides the help area list into several smaller sublists. The boundary indicator is "Y" (yes is always a border), "N" (no is never a border), or is in the range of 01 to 99 (a boundary when the indicator is on in the RPG program). The default is "N."

Runtime Global Help Area List Management

RSI maintains a global help area list. As different application forms are displayed, entries are added and deleted from the global help area list. When a new application form is displayed, the help area list is condensed by deleting all help areas that are completely within the area of the screen that is cleared by the new application form. Any null help areas that directly follow a deleted entry are also deleted. If an application form clears all 24 lines, the entire global help area list is deleted. If an application form does not clear any lines, no entries are deleted. After the global help area list is condensed, the help area list for the new application form is appended to the end of the global help area list. If the new application form is a variable starting line number form, the row numbers in the form's help area list are adjusted for the actual starting line number. Any help areas that would not be on the physical screen are deleted from the global help area list. The entire global help area list is scanned for entries that have the suppress selection indicator set on (by the RPG program); if any are found they are deleted from the global help area list.

15 Chapter 4 A Quick Reference to SIGEDITOR

Introduction

This chapter provides a quick guide to specific tasks of SIGEDITOR and gives step-by-step directions to perform a task. You can use these directions without reading chapters 1 and 2.

The tasks in this chapter are:

- Adding help to an existing form.
- Copying a field.
- Copying a form between forms files.
- Creating a field in an existing form.
- Deleting a field.
- Deleting a form.
- Drawing lines on a form.
- Editing attributes of a field.
- Editing form attributes.
- Erasing lines on a form.
- Generating RPG I/O specifications.
- Lengthening a field.
- Listing forms.
- Printing form and field attributes.
- Selecting a forms file.
- Shortening a field.

NOTE **NOTE** Use the cursor control keys to position the cursor when performing the tasks described in this chapter.

Adding Help to an Existing Form

1 From the main menu, press f1 (SELECT FRMS FIL).

2 Enter form file name and press ENTER .

3 Enter new help form name and press f1 (EDIT FORM).

4 Enter "Y" to create form.

5 Enter new help text and press ENTER . (Do not use RETURN or ENTER keys to move cursor around on screen; use arrow keys.)

6 Press ENTER to save text entry.

7 Press ENTER again if form name displayed is correct. If form name is not correct, enter the correct form name and press ENTER.

8 If there is another help form to create, press f1 (EDIT FORM) and go back to step 3. If there are no other forms to create, continue to step 9.

9 Press f7 (FORM UTIL).

10 Press f1 (HELP AREAS).

11 Enter forms file name if forms file name displayed on screen is not correct.

12 Enter form name which help form is to be attached to and press f2 (CREATE AREAS).

13 Move cursor to the upper left corner of help area and press f1 (UPPER LEFT).

14 Move cursor to the lower right corner of the help area and press f2 (LOWER RIGHT).

15 Press f3 (HELP ATTR).

16 Enter name of help form created earlier that corresponds with the help area you have just created and press ENTER .

17 To create additional help areas, go back to step 12. If you are finished creating help areas on this form, continue with the next step.

18 Press f8 (HELP MENU).

19 Press f8 (UTIL MENU).

20 Press f8 (EXIT FORM UTIL).

21 Press f8 (EXIT).

Copying a Field

1 From the main menu, press f1 (SELECT FRMS FIL).

2 Enter a valid forms filename and press ENTER .

3 Enter a valid form name and press f1 .

4 Press f8 to switch menus.

5 Position the cursor in the field to be copied.

6 Press the f1 key (HOLD FIELD).

7 Position the cursor where the field should be reproduced.

8 Press the f5 key (PLACE HOLD).

9 Press ENTER .

10 Press ENTER to save the form, or press f7 to abandon changes.

11 Press f8 (MAIN MENU) to return to the main menu.

Copying a Form Between Forms Files

1 From the main menu, press f1 (SELECT FRMS FIL) to open a forms file.

2 Enter a valid forms filename and press ENTER .

3 Press f7 (FORM UTILS).

4 Press f1 (COPY FORM).

5 Enter the names of the form and forms file using the tab key to move from field to field.

6 Press ENTER to copy the form.

7 Press f8 to return to the form utilities menu.

8 Press f8 to return to the edit main menu.

9 Press f8 to return to the main menu.

Creating a Field in an Existing Form

1 From the main menu, press f1 (SELECT FRMS FIL) to open a forms file.

2 Enter a valid forms filename and press ENTER .

3 Enter a valid form name and press f1 (EDIT FORM).

4 Position the cursor where the field is to begin.

5 Press f1 (CREATE FIELD) to create a new field.

6 Position the cursor where the field is to end.

7 Press f2 (STOP FIELD) to end the field.

8 Press ENTER .

9 Press ENTER again to save the form, or press f7 to abandon changes.

10 Press f8 to return to the main menu.

Deleting a Field

1 Press f1 (SELECT FRMS FIL) from the main menu.

2 Enter a valid forms filename and press ENTER .

3 Enter a valid form name and press f1 (EDIT FORM).

4 Press f8 (SWITCH MENU) to change menus.

5 Position the cursor in the field to be deleted.

6 Press f3 (DELETE FIELD).

7 Press ENTER

8 Press ENTER again to save the form, or press f7 to abandon changes.

9 Press f8 to return to the main menu.

Deleting a Form from a Forms File

1 From the main menu, press f1 (SELECT FRMS FIL) to open a forms file.

2 Enter a valid forms filename and press ENTER .

3 Press f7 (FORM UTILS).

4 Press f3 (DELETE FORM).

5 Enter the name of the form and press ENTER .

6 Press f8 to return to the form utilities menu.

7 Press f8 to return to the edit menu.

8 Press f8 to return to the main menu.

Drawing Lines on a Form

1 From the main menu, press f1 (SELECT FRMS FIL) to open a forms file.

2 Enter a valid forms filename and press ENTER .

3 Enter a valid form name and press f1 (EDIT FORM).

4 Press f8 (SWITCH MENU) **twice** to display the line drawing menu.

5 Press f3 (HORIZON SINGLE) or (VERTICAL SINGLE) to set the correct line type.

6 Position the cursor at the line's starting position.

7 Press f1 (FROM POSITION).

8 Position the cursor at the line's ending position.

9 Press f2 (TO POSITION). (Repeat steps 5 through 9, as needed, to draw all the desired lines.)

10 Press ENTER .

11 Press ENTER again to save the form, or press f7 to abandon changes.

12 Press f8 to return to the main menu.

Editing Field Attributes

1 From the main menu, press f1 (SELECT FRMS FIL) to open a form file.

2 Enter a valid forms filename and press ENTER .

3 Enter a valid form name and press f3 (EDIT FLD ATTRIB.).

4 Position the cursor in the correct field using the f1 , f2 , or f3 key.

5 To display the attribute specification form, press f7 (SEE ATTRIB.).

6 Press f7 (SEE ATTRIB) to display the attribute specification form.

7 Enter the attributes using the tab key to move from field to field.

8 Press f1 (SET FIELD).

9 Press f8 to return to the main menu.

Editing Form Attributes

1 Press f1 (SELECT FRMS FIL) from the main menu.

2 Enter a valid forms filename and press ENTER .

3 Enter a valid form name and press f2 to edit form attributes.

4 Enter specifications using the tab key to move from field to field.

5 Press ENTER when form specifications are complete.

6 Press f8 to return to the main menu.

Erasing Lines on a Form

1 Press f1 (SELECT FRMS FIL) from the main menu.

2 Enter a valid forms filename and press ENTER .

3 Enter a valid form name and press f1 (EDIT FORM).

4 Press f8 twice to display the line drawing menu.

5 Press the f5 key (ERASE).

6 Position the cursor at the line's starting position.

7 Press the f1 key (FROM POSITION).

8 Position the cursor at the line's ending position.

9 Press the f2 key (TO POSITION).

10 Press ENTER

11 Press ENTER again to save the form, or press f7 to abandon changes.

12 Press f8 to return to the main menu.

NOTE When deleting lines from a form, the line connectors (T's, crosses) will not be deleted. To delete them, re-draw a line (or portion of a line) over the area containing the connector.

Generating RPG I/O Specs

1. From the main menu, press f6 (GEN I/O SPECS).
2. Enter the name of the forms file and the file to contain RPG input and output specifications. If the file already exists, SIGEDITOR asks if the file can be overwritten.
3. Press the ENTER key.
4. Press f8 to return to the main menu.

NOTE This function creates input and output specifications for all forms in the open forms file.

Lengthening a Field

1. Press f1 (SELECT FRMS FIL) from the main menu.
2. Enter a valid forms filename and press ENTER .
3. Enter a valid form name and press f1 (EDIT FORM).
4. Press f8 to switch menus.
5. Position the cursor in the field to be lengthened.
6. Press f1 (ADD TO FIELD) until the field is the correct length.
7. Press ENTER .
8. Press ENTER again to save the form, or press f7 to abandon changes.
9. Press f8 to return to the main menu.

Listing Forms

1. Press f1 (SELECT FRMS FIL) from the main menu.
2. Enter a valid forms filename and press ENTER .
3. Press f1 to switch the list format.
4. Use f5 and f6 to page up and down through the forms list.
5. Press f8 to return to the main menu.

Printing Form and Field Specifications

1. From the main menu, press f5 (PRINT SPECS).
2. Enter the name of the forms file and the form and press ENTER .
3. Press f8 to return to the main menu.

Selecting a Forms File

1. From the main menu, press f1 (SELECT FRMS FIL).
2. Enter the forms filename and press ENTER .

Shortening a Field

1. Press f1 (SELECT FRMS FIL) from the main menu.
2. Enter a valid forms filename and press ENTER

3 Enter a valid form name and press f1 (EDIT FORM).

4 Press f8 to switch menus.

5 Position the cursor in the field to be shortened.

6 Press f2 (SHORTEN FIELD) until the field is the correct length.

7 Press ENTER .

8 Press ENTER again to save the form, or press f7 to abandon changes.

9 Press f8 to return to the main menu.

Updating/Deleting a Help Area

1 From the main menu, press f7 (FORM UTILS).

2 Press f1 (HELP AREAS).

3 Enter the forms filename and the application form name, then press f1 (UPDT/DEL AREAS).

4 Find the help area that you want to update/delete using the f5 and f6 (SCROLL UP/SCROLL DOWN) keys.

5 Enter the ID of the area that you want to update in the Help List ID field and press f1 (CHANGE ATTR).

6 Change the fields and press ENTER to set the new values.

7 Press f8 (HELP MENU).

8 Press f8 (UTIL MENU).

9 Press f8 (EXIT FRM UTILS).

10 Press f8 (EXIT).

16 Chapter 5 Using SIGEDITOR with RPG

Introduction

This chapter explains how to design SIGEDITOR forms for use with RPG. The chapter assumes you are familiar with SIGEDITOR. For more information about SIGEDITOR functions, refer to chapters 1 through 3.

Sample RPG Program

The program in Figure 4-1 uses two forms, FORM21 and FORM22, in the forms file SAMPLIB. The program adds, deletes, and updates records in a KSAM file.

```
|                                                                                 |
|                1         2         3         4         5         6         7    |
|            678901234567890123456789012345678901234567890123456789012345678901234|
|                                                                                 |
|                                                                                 |
|            HDUMPFILE JF           X     B       B N     P1 1                    |
|                                                                                 |
|        1   FWORKSTN UP  V       90           WORKSTNR     B                     |
|            F                                               KFORMS SAMPLIB       |
|            FCUSTMASTUC  F      128R 9AI    1 DISK                       A       |
|            F                                               KKEYFL KCUSTMST      |
|            F*                                                                   |
|            F* SAMPLE RSI PROGRAM USING THE WORK STATION AS A PRIMARY FILE       |
|            F*                                                                   |
|                                                                                 |
|        2   IWORKSTN NS  01   1 C1                                               |
|            I* FORM21                                                            |
|            I                                        1   1 FLD1                  |
|            I                                        2  100SSN                   |
|            I        NS  02   1 C2                                               |
|            I* FORM22                                                            |
|            I                                        1   1 FLD1                  |
|            I                                        2  27 NAME                  |
|            I                                       28  63 ADDR                  |
|            I                                       64  720ZIP                   |
|            I                                       73  82 PHONE#                |
|            I        NS                                                          |
|        3   ICUSTMASTNS                                                          |
|            I                                        1   90SSN                   |
|            I                                       10  35 NAME                  |
|            I                                       36  71 ADDR                  |
|            I                                       72  800ZIP                   |
|            I                                       81  90 PHONE#                |
|                                                                                 |
|            C*  CLEAN UP                                                         |
|        4   C   01                SETOF                     808182               |
|            C   01                SETOF                     656070               |
|            C*  DETERMINE ACTION REQUESTED                                       |
|        5   C   KA                SETON                     80  ADD RECORD       |
|            C   KB                SETON                     81    DEL RECORD     |
|            C   KC                SETON                     82    UPD RECORD     |
|            C   KG                SETON                     LR    END JOB        |
|            C   KG                GOTO EOF                                       |
|        6   C  N01                GOTO EOF                                       |
|            C*  READ RECORD FROM CUSTOMER MASTER FILE USING SSN AS A KEY         |
|            C                     SETOF                     42                   |
|            C           SSN       CHAINCUSTMAST             42    NO HIT         |
```

Figure 5-1. RPG Program (continued)

```
|                                                                                 |
|                1         2         3         4         5         6         7    |
|            678901234567890123456789012345678901234567890123456789012345678901234|
|                                                                                 |
|            C*  SET INDICATORS FOR ERRORS                                        |
|        7   C      80N42          SETON                     6070                 |
|            C     N80 42          SETON                     6065                 |
|            C           EOF       TAG                                            |
|        8   OWORKSTN D        1P                                                 |
|            O       OR        02                                                 |
|            O       OR        01 60                                              |
|            O       OR        01 81                                              |
|            O                                   K6 "FORM21"                      |
|            O                    70              6 "0001 1"                      |
|            O                    65              6 "0002 1"                      |
|            O**                                                                  |
|        9   O        D        01 80 42                                           |
|            O       OR        01 82N42                                           |
|            O                                   K6 "FORM22"                      |
|            O                         SSN        9                               |
|            O                         NAME      35                               |
|            O                         ADDR      71                               |
|            O                         ZIP       80                               |
|            O                         PHONE#    90                               |
|            O*  UPDATE DISK RECORD                                               |
|       10   OCUSTMASTD        02 82                                              |
|            O                         SSN    B   9                               |
|            O                         NAME   B  35                               |
|            O                         ADDR   B  71                               |
|            O                         ZIP    B  80                               |
|            O                         PHONE# B  90                               |
|            O*  ADD DISK RECORD                                                  |
|            O        DADD     02 80                                              |
|            O                         SSN    B   9                               |
|            O                         NAME   B  35                               |
|            O                         ADDR   B  71                               |
|            O                         ZIP    B  80                               |
|            O                         PHONE# B  90                               |
|            O*  DELETE DISK RECORD                                               |
|            O        DDEL     01 81N42                                           |
|            O                         SSN    B   9                               |
|            O                         NAME   B  35                               |
|            O                         ADDR   B  71                               |
|            O                         ZIP    B  80                               |
|            O                         PHONE# B  90                               |
|            O*  END OF SOURCE STATEMENTS                                         |
```

Figure 5-1. RPG Program (continued)

FORM21 (shown in Figure 4-2) allows the user to enter a social security number, choose to add a new record, delete an existing record, or update an existing record.

```
|                                                          |
|                        CUSTOMER INFORMATION FORM         |
|                                                          |
|                       Please enter Social Security Number|
|                                                          |
|                               306-46-7778                |
|                                                          |
|                                                          |
|                        RECORD ALREADY EXISTS             |
|                                                          |
|                                                          |
|                 Press CMD 1 TO ADD NEW RECORD            |
|                       CMD 2 TO DELETE EXISTING RECORD    |
|                       CMD 3 TO UPDATE EXISTING RECORD    |
|                       CMD 7 TO END JOB                   |
|                                                          |
|                                                           |
|                                                           |
|                                                           |
|                                                           |
|                                                           |
_____________________________________________________________
```

Figure 5-2. FORM21

If you choose to add a new record, the system searches the KSAM file for a record with the specified social security number. If the record already exists, an error message is displayed in a message field, as shown in Figure 4-2. If there is no associated record, FORM22 appears, and you can enter the record. The social security number appears on FORM22 automatically, as shown in Figure 4-3. When you press the ENTER key, the new record is added to the KSAM file.

```
_________________________________________________________________________________________
|                                                                                       |
|                                SOC. SEC. NO.  306-46-7778                             |
|                                                                                       |
|                                                                                       |
|                         NAME  MARY Q. QUAIL                                           |
|                                                                                       |
|                      ADDRESS  MORSE GROVE, FLA                                        |
|                                                                                       |
|                     ZIP CODE  339080000            PHONE NO.  813-672-1212            |
|                                                                                       |
|                                                                                       |
|                                     PRESS :ENTER:                                     |
|                                                                                       |
|                                                                                       |
|                                                                                       |
|                                                                                       |
|                                                                                       |
|     COMMAND    *PRINT      f3         f4           f5         f6         f7         f8|
|                SCREEN                                                                 |
|                                                                                       |
_________________________________________________________________________________________
```

Figure 5-3. FORM22

When a record is deleted, the system searches the KSAM file for that record. If found, the record is marked for deletion; if not found, a message is displayed.

When a record is updated, the record is displayed on FORM22 so modifications can be made. If the requested record cannot be found, a message is displayed.

Designing the Forms

FORM21 and FORM22 must be designed with SIGEDITOR. The final step when using SIGEDITOR is to generate the input/output specifications for the forms file containing the forms from the utilities menu (see chapter 3 for more details). Once this process is complete, you can begin constructing the RPG program.

FORM21

For this example, begin with FORM21 by setting the correct form attributes. Enable command keys 1, 2, 3, and 7 (A, B, C, and G) by placing a character in the space identified by the correct letter. The keys correspond to the three options available to the user (adding, deleting, or updating). Command key 7 is used to exit the program. Indicator 60 is used for override fields.

The visual design of FORM21 is achieved by typing screen constants (labels or directions) and creating three fields for the social security number, as shown in Figure 4-2. These fields are designated as numeric input fields and do not have to be output fields because they are only used to read the social security number. The three fields are inverse video fields. The RPG program reads these three fields as one.

Create a fourth field under the social security number fields to display any error messages. This is an output field conditioned by indicator 60 with the blink field attribute turned on. The field is only used to display a message to the user, not to return data to your program. The inverse video attribute is turned off, and the length of the field is the length of the message that is displayed. The message identification numbers are given in the program as constant values. The form title field on the first line is also a message type field; this one has the message identification (000301) coded on the form instead of the RPG program.

A fifth field, placed in the upper left corner of the form, is one character long. This field is a protected field of constant type, with its contents hidden from the user by the security video attribute. This field is specified as both input and output. The constant 1 is stored in the field and is used to identify the data from this form by the RPG program.

FORM22

The form attributes for FORM22 do not have to be changed because the form uses all default values.

The social security number fields are output fields, because they are used to display information only and not to return the information to your program. The inverse video attribute for these fields is off.

The NAME, ADDRESS, ZIP CODE, and PHONE NO. fields are input and output fields. These fields can return data to your program after an add or an update operation, or can display data to the screen for the user to modify. All fields are inverse video fields. The NAME and ADDRESS fields are specified as alphanumeric, and the ZIP CODE and PHONE NO. fields are numeric. The NAME, ADDRESS, and ZIP CODE fields are also specified as no-adjust and no-fill by leaving the adjust fill attribute blank; this causes the data in those fields to be left-justified.

FORM22 also has a one-character protected field of constant type in the upper left corner. The security video attribute is turned on. This field is both input and output and stores the value 2. The field is also used to identify data from this form by the RPG program.

Constructing the RPG Program

This section describes the relationship between SIGEDITOR and RPG. Although this section is not intended to be a detailed description of the sample program, it may be helpful to refer back to Figure 4-1 as you read this section. See the *RPG Reference Manual* for detailed information about RPG programming.

SIGEDITOR can generate most of the RPG specifications required to define the work station (RSI) file and its input/output records. These specifications can be included in your RPG program, with only minor editing. Figure 4-4 shows the specifications generated from FORM21 and FORM22. The sample program was constructed using these specifications. The forms filename, SAMPLIB, is specified on the "KFORMS" file continuation record. SIGEDITOR creates this specification for you. If that specification has been omitted, the forms filename would default to the program name (specified on the "H" specification or with

$CONTROL NAME=) suffixed with "FM". If the program name is missing, the default forms filename becomes "RPGOBJFM". The forms filename can be referenced by a file equation at runtime to redirect the forms file.

The social security number is split into three fields on the screen to allow for punctuation. In the RPG program, the three fields have been combined into a single field by modifying the "I" specifications so that it can be used as the key field for the CUSTMAST KSAM file. The phone number fields have also been combined into a single field to simplify the program.

The output specifications for the work station file were also modified to use the new field names for the social security number and the phone number. The error message (ERRMSG) field is used to pass the message identification (ID) to the form at runtime. The message ID's in this example are coded as output constants conditioned by indicators in the RPG program. They could be loaded into the field shown instead.

The RPG program can use the command key ("K") indicators to control the flow of the program as shown in the example. The command key indicators are all set off by RPG before each work station input operation. The user can turn on one command key indicator by pressing the command key associated with that indicator. The forms can use RPG indicators to control form and field attributes. The RPG program must set these indicators before the form is displayed so that the desired functions will be performed.

```
|                                                                                  |
|                1         2         3         4         5         6         7     |
|            678901234567890123456789012345678901234567890123456789012345678901234|
|                                                                                  |
|                                                                                  |
|            FWRKSTN  UD  V      6          WORKSTNR                              |
|            F                                           KFORMS SAMPLIB           |
|            IWRKSTN                                                              |
|            I*                                                                   |
|            I* FORM21                                                            |
|            I*                                                                   |
|            I                                     1   1 FLD1                     |
|            I                                     2   40SSN1                     |
|            I                                     5   60SSN2                     |
|            I                                     7  100SSN3                     |
|            I*                                                                   |
|            I* FORM22                                                            |
|            I*                                                                   |
|            I                                     1   1 FLD1                     |
|            I                                     2  27 NAME                     |
|            I                                    28  63 ADDR                     |
|            I                                    64  720ZIP                      |
|            I                                    73  75 PN1                      |
|            I                                    76  78 PN2                      |
|            I                                    79  82 PN3                      |
|            OWRKSTN                                                              |
|            O*                                                                   |
|            O  FORM21                                                             |
|            O*                                                                    |
|            O                                K6 'FORM21'                          |
|            O                      ERRMSG     6                                   |
|            OWRKSTN                                                               |
|            O*                                                                    |
|            O* FORM22                                                             |
|            O*                                                                    |
|            O                                K6  'FORM22'                         |
|            O                      PT1        3                                   |
|            O                      PT2        5                                   |
|            O                      PT3        9                                   |
|            O                      NAME      35                                   |
|            O                      ADDR      71                                   |
|            O                      ZIP       80                                   |
|            O                      PNPT1     83                                   |
|            0                      PNPT2     86                                   |
|            O                      PNPT3     90                                   |
|                                                                                  |
```

Figure 5-4. RPG Specifications

Overlays and Variable Starting Lines

Overlays are two or more forms that are displayed so that portions of all the forms are visible at the same time. You can create overlays using the start line number and number of lines to clear form attribute entries.

For example, a form that starts on line 13 and clears 12 lines occupies the bottom half of the screen, leaving the top half of the prior form displayed unchanged. This type of overlay can be used to build a screen with several forms that can be combined dynamically at runtime. For example, this feature can be used in an inquiry application.

Another example is a form that does not clear any lines, in which case both forms are displayed at the same time. Input capable fields on the first form are no longer active after the second form is displayed. Input capable fields on the second form whose screen positions coincide with fields or constants from the first form have the field or constant as its initial value. This type of overlay can be used to create applications that use the output data from one form as input data to the next form with the combinations taking place dynamically at runtime, perhaps to model a new entry after an existing entry of a different type in a database.

Variable Starting Line forms are forms that start on a line number supplied by the RPG program at runtime. The start line number form attribute must be V, and the "KSTART" file continuation record must contain the name of a numeric field two positions long with zero decimal places.

Load the starting line number for the form into the field before you display the form. An example of this feature is a data entry application that uses a form that is one or two lines long. The same form can be displayed at different locations on the screen so that the user can see the last few transactions while entering the current transaction.

An example of overlays and variable start line forms used together is an order entry application that begins with a small form that occupies the top few lines of the screen to enter header information. The next form is a one-line form that is displayed once for each detail line of the order, moving down the screen a line at a time. The final form occupies the last line or two of the screen and contains the totals for the order. An entire order (assume the maximum detail lines is 15) can be seen at one time even though it was entered on many different forms.

I Appendix A SIGEDITOR Error Messages

Introduction

This appendix lists and explains the SIGEDITOR error messages.

MESSAGE UNABLE TO OPEN TERMINAL. (SIGERR 001)

CAUSE There is a system problem with the terminal.

ACTION Contact your system manager if you are unable to resolve this problem.

MESSAGE UNABLE TO OPEN LIBRARY FORM FILE. (SIGERR 002)

CAUSE MPE could not find the specified form file.

ACTION Verify the form filename.

MESSAGE FILE IS NOT A VALID FORM FILE OR IS CORRUPT. (SIGERR 003)

CAUSE An invalid form file was selected to be opened.

ACTION Verify the form filename.

MESSAGE FILE SPECIFIED IS NOT A VALID FORMS FILE. (SIGERR 004)

CAUSE An invalid form file (or an old forms file) was selected to be opened.

ACTION Verify the form filename. Verify that the form was created with version 7.0 or higher of SIGEDITOR.

MESSAGE UNABLE TO OPEN TEMPORARY WORK FILE. (SIGERR 005)

CAUSE There is either an existing temporary file or the system is out of memory.

ACTION Purge the temporary file if it exists.

MESSAGE UNEXPECTED FILE ERROR ON FORMS FILE. (SIGERR 007)

CAUSE There is a possible system problem.

ACTION Contact your system manager if you are unable to resolve this problem.

MESSAGE FORM DOES NOT EXIST. (SIGERR 011)

CAUSE The form file does not contain the specified form.

ACTION Verify that spelling of form name is correct, including the usage of upper and lower case.

MESSAGE FORM SPECIFIED ALREADY EXISTS. (SIGERR 013)

CAUSE The name specified for this form is the name of an existing form in the same forms file.

ACTION Choose a unique form name.

MESSAGE PLEASE ENTER A VALID FORM NAME. (SIGERR 015)

CAUSE The specified form name is not valid.

ACTION Use only alphanumeric characters when naming a form.

MESSAGE THIS FORM CONTAINS NO FIELDS FOR EDITING ATTRIBUTES. (SIGERR 016)

CAUSE You can only edit field attributes when fields exist within a form.

ACTION Check the form for fields.

MESSAGE FILE SPECIFIED ALREADY EXISTS. (SIGERR 019)

CAUSE The specified file currently exists as a permanent file.

ACTION Enter another filename.

MESSAGE UNABLE TO OPEN OUTPUT FILE SPECIFIED. (SIGERR 020)

CAUSE SIGEDITOR could not find the specified output file.

ACTION Verify that the filename is correct.

MESSAGE UNEXPECTED FILE SYSTEM ERROR ON OUTPUT FILE. (SIGERR 022)

CAUSE There is a possible system problem.

ACTION Contact your system manager if you are unable to resolve this problem.

MESSAGE UNABLE TO OPEN S AND D SPECIFICATIONS FILE. (SIGERR 025)

CAUSE SIGEDITOR could not find the specified file.

ACTION Verify that the filename is correct.

MESSAGE FORM FILE SPECIFIED ALREADY EXISTS. (SIGERR 026)

CAUSE The specified file currently exists as a permanent file.

ACTION Select a unique form filename.

MESSAGE UNABLE TO PURGE OLD FORMS FILE. (SIGERR 029)

CAUSE There is a system problem, or the file is open from another source.

ACTION Contact your system manager if you are unable to resolve this problem.

MESSAGE UNEXPECTED FILE SYSTEM ERROR ON S AND D SPEC FILE. (SIGERR 030)

CAUSE There is a possible system problem.

ACTION Contact your system manager if you are unable to resolve this problem.

MESSAGE RPG VERSION 6.XX FORMS FILE. RECONVERT S&D SPECS (SIGERR 031)

CAUSE SIGEDITOR tried to access a forms file created with RPG version 6.XX.

ACTION Reconvert the S&D specifications using the current version of SIGEDITOR.

MESSAGE FILE SPECIFIED IS NOT AN OLD FORMS FILE. (SIGERR 032)

CAUSE An invalid form file (or an old forms file) was selected.

ACTION Verify the form filename. Verify that the form was created with version 7.0 or higher of SIGEDITOR.

MESSAGE UNABLE TO OPEN MESSAGE CATALOG FILE. (SIGERR 035)

CAUSE MPE could not find the specified catalog.

ACTION Contact your system manager if you are unable to resolve this problem.

MESSAGE UNKNOWN COMMAND ARGUMENT PASSED TO PROGRAM. (SIGERR 038)

CAUSE An invalid parameter was passed to SIGEDITOR.

ACTION Refer to the descriptions of parameters accepted by SIGEDITOR (in this manual).

MESSAGE UNEXPECTED FILE SYSTEM ERROR HAS OCCURRED. (SIGERR 039)

CAUSE There is a possible system problem.

ACTION Contact your system manager if you are unable to resolve this problem.

MESSAGE INVALID PROCESSING OPTION SPECIFIED. (SIGERR 040)

CAUSE A value other than "A" or "P" was input.

ACTION Type "A" to process all form files or "P" for a prompt before processing each forms file.

MESSAGE A (Y)ES OR (N)O RESPONSE IS REQUIRED. (SIGERR 041)

CAUSE A value other than "Y" or "N" was input while using the forms conversion program.

ACTION Type "Y" to review conversion results, or "N" to omit the review step.

MESSAGE INVALID FILE SETS SPECIFIED. SEE MANUAL. (SIGERR 043)

CAUSE An invalid file set was specified.

ACTION See the rules for valid file sets when converting S and D specifications (chapter 2).

MESSAGE FILE SYSTEM ERROR ON OUTPUT FILE. (SIGERR 047)

CAUSE There is a possible system problem.

ACTION Contact your system manager if you are unable to resolve this problem.

MESSAGE UNEXPECTED FILE SYSTEM ERROR ON CONSOLE WORK FILE. (SIGERR 049)

CAUSE There is a possible system problem.

ACTION Contact your system manager if you are unable to resolve this problem.

MESSAGE FIELDS AND CONSTANTS HAVE EXCEEDED CAPACITY. (SIGERR 050)

CAUSE There are more than 256 fields and constants on this form.

ACTION Reduce the number of fields and constants to less than 256.

MESSAGE EXTRA DATA SEGMENT GET/FREE FAILURE. (SIGERR 057)

CAUSE A system error has occurred.

ACTION Contact your system manager for assistance.

MESSAGE EXTRA DATA SEGMENT READ/WRITE FAILURE (SIGERR 058)

CAUSE A system error has occurred.

ACTION Contact your system manager for assistance.

MESSAGE MAXIMUM NUMBER OF 10 FORMS EXCEEDED. (SIGERR 059)

CAUSE There are too many record definitions for the console file.

ACTION Reduce the number of record types in the RPG input specifications for this file.

MESSAGE TOTAL NUMBER OF FIELDS WILL NOT FIT ON FORM. (SIGERR 060)

CAUSE There are too many fields defined for a console form.

ACTION Reduce the number of fields defined in the RPG input specifications for this form.

MESSAGE NON-NUMERIC ID WAS ENTERED. (SIGERR 061)

CAUSE A special or alphabetic character was entered for either the "LIST ENTRY ID" or "NEW LIST ENTRY ID" field on the help area update/delete form.

ACTION Reenter the ID using only the digits 0-9. Leading or trailing blanks are also allowed.

MESSAGE ID ENTERED WAS NOT FOUND. (SIGERR 062)

CAUSE The "LIST ENTRY ID" entered on the help area update/delete form does not exist.

ACTION Recheck the ID to make sure that it exists and was entered correctly.

MESSAGE NEW ID ENTERED ALREADY EXISTS. (SIGERR 063)

CAUSE The "NEW LIST ENTRY ID" entered on the help area update/delete form for a change ID operation already exists.

ACTION Reenter the ID, making sure that it has not already been used.

MESSAGE ZERO (0) IS NOT A VALID ID. (SIGERR 064)

CAUSE Either the "LIST ENTRY ID" of the "NEW LIST ENTRY ID" field was left blank or contains all zeros on the help area update/delete form.

ACTION Reenter the ID using a valid number.

MESSAGE ID ENTERED IS NOT ON THE SCREEN. (SIGERR 065)

CAUSE The "LIST ENTRY ID" entered on the help area update/delete form for a change attributes operation is not on the screen.

ACTION Reenter the ID using a number that appears on the screen, or use the scrolling functions to display the entry you want to change.

MESSAGE HELP FORM NAME IS INVALID. (SIGERR 066)

CAUSE The help form name entered is not in the required format.

ACTION Reenter the name, making sure that it is 8 characters long and that the last 2 characters are numeric digits from 0 through 9.

MESSAGE SUPPRESS SELECTION IND. MUST BE BLANK OR 01 TO 99. (SIGERR 067)

CAUSE The suppress selection indicator is not a legal value.

ACTION Reenter the indicator, making sure it is blank or an indicator in the range 01 to 99.

MESSAGE NON-NUMERIC HELP DOC. STARTING LINE NUMBER ENTERED. (SIGERR 068)

CAUSE A numeric value between one and 24 must be used here.

ACTION Reenter the appropriate data.

MESSAGE UPPER LEFT ROW MUST BE ZERO OR 01 TO 24. (SIGERR 069)

CAUSE The upper left row number is not in the required range.

ACTION Reenter the row number, making sure that it is in the range 00 to 24.

MESSAGE UPPER LEFT COLUMN MUST BE ZERO OR 01 TO 80. (SIGERR 070).

CAUSE The upper left column number is not in the required range.

ACTION Reenter the column number making sure that it is in the range 00 to 80.

MESSAGE LOWER RIGHT ROW MUST BE ZERO OR 01 TO 24. (SIGERR 071)

CAUSE The lower right row number is not in the required range.

ACTION Reenter the row number, making sure that it is in the range 00 to 24.

MESSAGE LOWER RIGHT COLUMN MUST BE ZERO OR 01 TO 80. (SIGERR 072)

CAUSE The lower right column number is not in the required range.

ACTION Reenter the column number, making sure that it is in the range 01 to 80.

MESSAGE INVALID NULL HELP AREA DEFINITION. (SIGERR 073)

CAUSE One or more row or column numbers in the help area definitions are not zero for a null help area definition.

ACTION Reenter the row and column numbers as zeros to define a null help area or as nonzero values in the proper ranges to define a normal help area.

MESSAGE UPPER LEFT IS NOT ABOVE AND TO THE LEFT OF LOWER RIGHT. (SIGERR 074)

CAUSE The upper left row and column numbers used to define the help area must be less than or equal to the lower right row and column numbers, and they are not.

ACTION Reenter the row and column numbers, making sure that they are correct in relation to each other.

MESSAGE UPPER LEFT MUST BE DEFINED FIRST. (SIGERR 075)

CAUSE An attempt was made to select the lower right corner of the help area before the upper left corner was selected.

ACTION Select the upper left corner first, then the lower right corner.

MESSAGE UPPER LEFT AND/OR LOWER RIGHT CORNER NOT DEFINED. (SIGERR 076)

CAUSE An attempt was made to set the attributes for a help area before both corners were selected.

ACTION First select the upper left corner of the help area, then select the lower right corner of the help area, then select the help area attributes.

MESSAGE MAXIMUM NUMBER OF HELP AREAS (256) ALREADY EXIST. (SIGERR 077)

CAUSE The maximum number of help areas have already been defined for this form.

ACTION You must delete some help areas before any more can be defined for this form. Perhaps several help areas can be combined into one larger help area.

MESSAGE UPPER LEFT ROW CANNOT BE LESS THAN START LINE. (SIGERR 078)

CAUSE A help area must start at or below the starting line of the application form.

ACTION Redefine the corners of the help area so that the help area location is correct in relation to the starting line number of the application form.

J Appendix B Summary of the SIGEDITOR Keys

Introduction

This appendix summarizes the function of each key.

Main Menu

SELECT	CONVERT	MERGE	PRINT	GEN I/O	FORM	EXIT
FRMS FIL	S&D SPEC	CATALOG	SPECS	SPECS	UTILS	

Key	Description
SELECT FRMS FIL	Opens the file once a forms file has been named.
CONVERTS &D SPEC	Converts files containing S, D, and H specifications from an IBM system to the files used by SIGEDITOR.
MERGE CATALOG	Merges a catalog file with a forms file.
PRINT SPECS	Prints a copy of the form and field specifications.
GEN I/O SPECS	Generates the I/O specifications of all the forms in a single forms file.
FORM UTILS	Switches to the Form Utilities menu.
EXIT	Exits the SIGEDITOR program.

Edit Menu

EDIT	EDT FORM	EDT FLD	SWITCH	PREV PAG	NEXT PAG	FORM	MAIN
FORM	ATTRIB	ATTRIB	LST FMT	FRM LIST	FRM LIST	UTILS	MENU

Key	Description
EDIT FORM	Takes you to the detailed form editing keys.
EDT FORM ATTRIB	Takes you to the Edit Form Attributes screen to enter form attributes.
EDT FLD ATTRIB	Takes you to the field attribute keys.
SWITCH LST FMT	Changes the form listing format on the Edit Menu.
PREV PAG FRM LIST	Displays the previous page of the forms list.
NEXT PAG FRM LIST	Displays the next page of the forms list.
FORM UTILS	Takes you to the Form Utilities Menu.
MAIN MENU	Returns you to the Main Menu.

Help Areas Menu

```
    CREATE     UPDT/DEL          UTIL
    AREAS        AREAS           MENU
```

Key	**Description**
CREATE AREAS	Takes you to the create help areas keys.
UPDT/DEL AREAS	Takes you to the update/delete help areas keys
UTIL MENU	Takes you to the form utilities menu.

Form Utilities Menu

```
 HELP      DELETE  COPY  MERGE    PRINT  GEN I/O  EXIT FRM
 AREAS       FORM   FORM  CATALOG  SPECS  SPECS    UTILS
```

Key	**Description**
HELP AREAS	Takes you to the help areas menu
DELETE FORM	Deletes a specified form.
COPY FORM	Creates a duplicate of a specified form.
MERGE CATALOG	Merges a forms file with a catalog file.
PRINT SPECS	Prints a copy of the form and field specifications.
GEN I/O SPECS	Generates the I/O specifications for all the forms in a single forms file.
EXIT FRM UTILS	Exits the Form Utilities Menu and returns you to the Edit Menu.

Keys for Creating Help Areas

```
 UPPER  LOWER  HELP      REFRESH  SHOW ALL  ABANDON   HELP
 LEFT    RIGHT  ATTR      FORM      AREAS     ALL CHGS  MENU
```

Key	Description
UPPER LEFT	Selects the upper left corner of an imaginary rectangle that defines the help area.
LOWER RIGHT	Selects the lower right corner of an imaginary rectangle that defines the help area. The help area is outlined on the screen after the lower right corner has been selected, except if the help area is only a single position.
HELP ATTR	Displays the help attribute form after the corners have been selected.
REFRESH FORM	Redraws the application (user) form on the screen without any help areas outlined.
SHOW ALL AREAS	Outlines all help areas defined for this form on the screen.

Key	Description
ABANDON ALL CHGS	Discards all help areas defined during this create session and returns you to the help areas menu.
HELP MENU	Saves all help areas defined during this create session and returns you to the help areas menu.

Keys for Designing a New Form

START	STOP	CENTER		REFRESH	SWITCH
FIELD	FIELD	LINE			MENU

Key	Description
START FIELD	Identifies the start of a new field.
STOP FIELD	Identifies the end of the field.
CENTER LINE	Centers a line of the form.
REFRESH	"Repaints" the form including all changes up to and including the last function key pressed.
SWITCH MENU	Takes you to the detailed form editing keys when designing a form.

Keys for Detailed Form Editing

ADD TO	SHORTEN	DELETE	HOLD	PLACE	CLEAR	REFRESH	SWITCH
FIELD	FIELD	FIELD	FIELD	FIELD	HOLD		MENU

Key	**Description**
ADD TO FIELD	Lengthens a field one position each time the key is pressed.
SHORTEN FIELD	Shortens or removes spaces from a field; one space is removed each time the key is pressed.
DELETE FIELD	Deletes an entire field.
HOLD FIELD	Copies or moves a field from one position to another.
PLACE FIELD	Places the contents of the held field onto the form.
CLEAR HOLD	Clears the currently held field.
REFRESH	"Repaints" the form including all changes up to and including the last function key pressed.
SWITCH MENU	Takes you to the line drawing keys when designing a form.

Keys for Drawing Lines

FORM	TO	HORIZON	VERTICAL	ERASE	REFRESH	SWITCH
POSITION	POSITION	SINGLE	SINGLE			MENU

Key	Description
FROM POSITION	Identifies the beginning of the line.
TO POSITION	Identifies the end of the line.
HORIZON SINGLE	Specifies the type of horizontal lines; can be SINGLE, BOLD, or DOUBLE.
VERTICAL SINGLE	Specifies the type of vertical lines; can be SINGLE, BOLD, or DOUBLE.
ERASE	Takes you to the Erase keys in order to erase lines from a form.
REFRESH	"Repaints" the form including all changes up to and including the last function key pressed.
SWITCH MENU	Returns you to the form design keys (START FIELD, STOP FIELD, etc.).

Erase Keys

ERASE	ERASE	EDIT
FROM	TO	MENU

Key	Description
ERASE FROM	Indicates the beginning of the line to be erased.
ERASE TO	Indicates the end of the line to be erased.
EDIT MENU	Returns you to the Edit Menu after using Erase keys.

Field Attribute Keys

NEXT	PREV	FIRST	SET	DEFAULT	SEE/SET	EDIT
FIELD	FIELD	FIELD	FIELD	ATTRIB.	ATTRIB.	MENU

Key	Description
NEXT FIELD	Moves the cursor to the next field in the form.
PREV FIELD	Moves the cursor to the previous field in the form.
FIRST FIELD	Moves the cursor to the first field in the form.
SET FIELD	Keeps any changes made to the attributes.
DEFAULT ATTRIB	Sets all field attributes back to their original default values.
SEE/SET ATTRIB	Displays the current attributes of the field identified by the cursor.

Key	Description
EDIT MENU	Returns you to the Edit Menu.

Update/Delete Help Areas

CHANGE DELETE CHANGE SHOW ALL SCROLL SCROLL ABANDON HELP
ATTR AREA ID AREAS FRWD BACK ALL CHGS MENU

Key	Description
CHANGE ATTR	Allows you to change the help attributes of a specific help area displayed on the screen.
DELETE AREA	Deletes (removes) a specific help area.
CHANGE ID	Allows you to rearrange the help areas using their ID numbers.
SHOW ALL AREAS	Outlines all help areas defined for this form on the screen.
SCROLL FRWD	Displays the next page (12 lines) of help areas on the screen when there are move than 12 help areas defined for a form.
SCROLL BACK	Displays the previous page (12 lines) of help areas on the screen when there are more than 12 help areas defined for a form.
ABANDON ALL CHGS	Discards all changes to help areas made during this update/delete session and returns you to the help areas menu.
HELP MENU	Saves all changes to help areas made during this update/delete session and returns you to the help areas menu.

Other Keys

RETURN ABANDON ABANDON
TO EDIT EDITS

Key	Description
RETURN TO EDIT	Returns you to the previous form editing screen
ABANDON EDITS	Exits from the Edit Form function without saving changes.
ABANDON	Exits the current function without saving the changes and returns you to the previous menu.

K Appendix C Calculating the Self-Check Digit

Introduction

As described in chapter 2, the self-check digit is used for numeric fields. If a mode of self-checking is specified, the field is checked by the checking algorithm when information is entered.

Modulus 10 Self-Check Digit

To calculate the modulus 10 self-check digit, follow these steps:

1 Multiply the rightmost position and every other position, moving to the left, of the number by 2.

Example: If your number is 1 2 3 4 5 6 7, multiply the digits 7, 5, 3, and 1 by 2 to get the numbers 14, 10, 6, and 2.

2 Add the digits of the products obtained from step 1 above to the digits that were not multiplied.

Example: The products obtained from step 1 above are 14, 10, 6, and 2; the digits that were not multiplied are 2, 4, and 6. Add all these digits: 1 + 4 + 1 + 0 + 6 + 2 + 2 + 4 + 6 = 26.

3 Subtract the value obtained in step 2 from the next higher multiple of 10; the result is the self-check digit.

Example: The next higher multiple of 10 from the result 26 obtained in step 2 above is 30. Subtracting 26 from 30 gives the result of 4, which is the self-check digit.

4 The check digit is then appended to the right end of the number to create the final result.

Modulus 11 Self-Check Digit

To calculate the modulus 11 self-check digit, follow these steps:

1 Assign a multiplication factor to each digit position of your number as follows: starting with the rightmost position of your number, assign the numbers 2, 3, 4, 5, 6, 7, 2, 3, 4, 5, 6, 7, 2,...from right to left.

Example: If your number is 1 2 3 4 5 6 7, the multiplication factor is assigned as follows:

Your number:	1	2	3	4	5	6	7
Multiplication factor:	2	7	6	5	4	3	2

2 Multiply each digit by its multiplication factor.

Example:

Your number:	1	2	3	4	5	6	7
Multiplication factor:	2	7	6	5	4	3	2
Results after multiplying:	2	14	18	20	20	18	14

3 Add the products from the multiplication.

Example: 2 + 14 + 18 + 20 + 20 + 18 + 14 = 106

4 Divide the sum from step 3 by 11.

Example: 106 divided by 11 is 9 with a remainder of 7.

If the remainder is 0, the self-check digit is 0. If the remainder is 1, your number does not have a modulus 11 self-check digit, and you should not use the number in the fields you define with modulus 11 self-check fields.

5 Subtract the remainder obtained in step 4 from 11 to obtain the self-check digit.

Example: The remainder from step 4 is 7; 11 minus 7 is 4, which is the self-check digit.

HP RPG/XL Utilities-Part 4 RPGINIT

Preface to RPGINIT

This publication is the reference manual for Hewlett-Packard's RPGINIT. RPGINIT is a utility used by RPG programmers to create and initialize a Local Data Area file, create and initialize the RPG user date file, and display the contents of both files.

This manual assumes that you are an experienced RPG programmer using RPGINIT.

Additional Documentation

More information on RPG/3000 can be found in the *RPG/3000 Reference Manual*.

17 Chapter 1 RPGINIT

RPGINIT performs the following:

- Creates a **Local Data Area** (LDA) file and initializes the file to ASCII blanks.
- Creates and initializes the RPG user date file RPGUDATE to the current system date.
- Displays the contents of both files and the session's user switch settings.

Creating and Initializing the Local Data Area

The **Local Data Area** is a temporary file named LDAFILE and is used to pass information between RPG programs and other programs or job control procedures (such as PROCMON/3000). RPGINIT creates and initializes LDAFILE to blanks. The file can be created as 1 through 32 multiples of 256 bytes (such as 256, 512, 768, 1024,..., 8192 bytes). To access the **Local Data Area** from an RPG program, add an Input specification with a "U" in column 18 and a "DS" in columns 19 through 20. This defines a data structure named "LDA"; specification of the data structure name is optional.

The LDA record layout is defined in the Input specifications following the "UDS" specification. The runtime library defines the entire **Local Data Area** as an array "LDA". The array is made up of 1-byte elements totalling the size of the entire LDAFILE (that is, 256, 512, 768, 1024,..., 8192 bytes). The LDA array is defined by RPG; do not define the array on an Extension specification.

When the RPG program begins, the **Local Data Area** is automatically read into the LDA data structure. The data is available for first cycle (1P) output. When the program terminates, the LDA data structure is automatically written to LDAFILE.

NOTE If a file named LDAFILE already exists when RPGINIT is run, that file is reinitialized to blanks and any prior data is lost.

Creating and Initializing the RPG User Date File

RPGINIT creates and initializes the RPG User Date File, named RPGUDATE, to the current system date. RPGUDATE is a temporary file containing one 80-byte record.

The date is in the form *mmddyy*, beginning in position 1 of the RPGUDATE record. This file is used for RPG programs with an "F" specified in Header Column 17 (UDATE Source). Programs with this specification initialize the RPG reserved date fields UDATE, UDAY, UMONTH, and UYEAR from the RPGUDATE file rather than from the operating system date. This allows programs to be run using a date that is different than the operating system date.

Running RPGINIT

To create and initialize the files LDAFILE and RPGUDATE, run RPGINIT with the command

```
RUN RPGINIT.PUB.SYS [;PARM=n ] [,QUIET]
```

where *n* can be a value of 1 through 32. The PARM value *n* represents a multiple of 256, as shown in the

the table below:

Value of PARM	Action
1	Creates the LDAFILE with 256 bytes.
2	Creates the LDAFILE with 512 bytes.
3	Creates the LDAFILE with 768 bytes.
32	Creates the LDAFILE with 8192 bytes.

If a value for PARM is not specified or is less than 1, the value for PARM defaults to 2 (a 512-byte LDAFILE). If the value for PARM is greater than 32, the value defaults to 32 (an 8192-byte LDAFILE). For example, to create and initialize an RPGUDATE file and a 512 byte Local Data Area, use the following commands before accessing the LDA file:

```
RUN RPGINIT.PUB.SYS;PARM=2
```

or

```
RUN RPGINIT.PUB.SYS
```

For a 256-byte **Local Data Area**, use the following command:

```
RUN RPGINIT.PUB.SYS; PARM=1
```

The QUIET option allows you to suppress the display of informational messages when using RPGINIT to create and initialize LDAFILE and RPGUDATE. This can be useful when running RPGINIT from a User Defined Command (UDC) as shown in the example below. In most applications you can include the RPGINIT command in a logon User Defined Command, as shown in the following UDC file:

```
SETUP

OPTION LOGON

RUN RPGINIT.PUB.SYS, QUIET

:

<other commands for setting up>
```

Displaying Contents

To display the contents of the **Local Data Area** (LDAFILE), the RPGUDATE file, and the user switch settings (U1 through U8), run RPGINIT with the SHOW entry point, as shown below:

```
RUN RPGINIT.PUB.SYS,SHOW [;PARM=n ]
```

By running RPGINIT with the SHOW entry point, the current contents of the LDA file are displayed on the terminal. The character data is blocked in groups of ten characters for easy verification of correct updating of the LDA file by programs and by other job control procedures (such as PROCMON/3000). This screen is shown below:

```
HP32104A.07.00 RPG INITIALIZER UTILITY (RPGINIT) (C) HEWLETT-PACKARD CO. 1986
```

```
RPGUDATE: 090186
USWITCH: U1=OFF, U2= ON, U3=OFF, U4=OFF, U5=OFF, U6=OFF, U7=OFF, U8=OFF
LDAFILE:
0001: AA2001 AS SEM. BRACK  ET    22
0051:
0101:
0151:
0201:
0251:
MORE (Y/N)?
0257:
0307:
0357:
0407:
0457:
0507:
```

If a value for PARM is not specified or is less than 1, you are prompted to view one 256-byte LDA record at a time. If the value for PARM is greater than 1, the number specified, or the total number of records in LDAFILE (whichever is smaller) is displayed without any prompting. To reinitialize the LDA file to blanks and the RPG user date file to the system date, rerun RPGINIT with the same PARM value of 1 through 32.