



# RTE INTERACTIVE EDITOR

## Reference Manual



---

HEWLETT-PACKARD COMPANY  
11000 WOLFE ROAD, CUPERTINO, CALIFORNIA, 95014

Library Index Number  
2RTE.320.92060-90014

# PUBLICATION NOTICE

Information in this manual reflects the RTE Interactive Editor. Changes in text to document software updates subsequent to the initial release are supplied in manual update notices and/or complete revisions to the manual. The history of any changes to this edition of the manual is given below under "Publication History." The last change itemized reflects the software currently documented in the manual.

Any changed pages supplied in an update package are identified by a change number adjacent to the page number. Changed information is specifically identified by a vertical line (revision bar) on the outer margin of the page.

## PUBLICATION HISTORY

Second Edition . . . . . May 78 (Software Rev. Code 1805)  
Change 1 . . . . . June 78 (Software Rev. Code 1805)  
Change 2 . . . . . Dec 78 (Software Rev. Code 1901)

### NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

This document contains proprietary information which is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced or translated to another program language without the prior written consent of Hewlett-Packard Company.

**HP Computer Museum**  
**[www.hpmuseum.net](http://www.hpmuseum.net)**

**For research and education purposes only.**

# MANUAL CHANGE NOTICE

## MANUAL IDENTIFICATION

**Part Number:** 92060-90014

**Print Date:** May 1978

**Title:** RTE Interactive Editor  
Reference Manual

**Library Index No.:** 2RTE.320.92060-90014

## CHANGE IDENTIFICATION

**Change Number:** 2

**Print Date:** Dec 1978

**Software Revision:** 1901



## THE PURPOSE OF THIS MANUAL CHANGE

is to accumulate all changes to the current edition of the manual. Earlier changes, if any, are contained herein for your convenience. *(If you have made all previous changes to this manual, you need only make the changes described under the change number indicated above.)*

## CHANGED PAGES ARE IDENTIFIED

by the change number at the bottom of the page and a vertical line (change bar) in the outside margin to indicate the area of the text that has been changed..

## NEW PAGES ARE IDENTIFIED

by the change number at the bottom of the page. "New" pages are those which were not present when the current edition of the manual was published.

## TO UPDATE YOUR MANUAL

locate the Change Number, indicated above, on the back of this page and follow the instructions provided.



**TECHNICAL MANUAL CHANGES  
(92060-90014)**

**CHANGE 1:**

Reason for Change 1: Incorporate technical manual enhancements.

- A. Replace existing pages with enclosed updated pages.

**CHANGE 2:**

Reason for Change 2: Correct errors in printed manual.

- A. Page 1-5, sixth line from bottom of page:  
change ">0 read protected"  
to ">0 write protected" ✓
- B. Page 4-5, format box at top of page:
  - 1. Change second line  
from "Ts1,s2 . . . change tap stops, leave control character." ✓  
to "Txs1,s2 . . . change tap stops."
  - 2. Change third line  
from "Txs1,s2 . . . s10"  
to "Txs1,s2, . . . ,s10" ✓
- C. Replace title page, reader comment sheet, and backcover with new pages supplied.

# MANUAL CHANGE NOTICE

## MANUAL IDENTIFICATION

**Part Number:** 92060-90014

**Print Date:** May 1978

**Title:** RTE Interactive Editor  
Reference Manual

**Library Index No.:** 2RTE.320.92060-90014

## CHANGE IDENTIFICATION

**Change Number:** 1

**Print Date:** June 1978

**Software Revision:** 1805



## THE PURPOSE OF THIS MANUAL CHANGE

is to accumulate all changes to the current edition of the manual. Earlier changes, if any, are contained herein for your convenience. *(If you have made all previous changes to this manual, you need only make the changes described under the change number indicated above.)* This change notice may consist of changed pages, new pages, write-in change instructions, or a combination of all.

## CHANGED PAGES ARE IDENTIFIED

by the change number at the bottom of the page and a vertical line (change bar) in the outside margin to indicate the area of the text that has been changed.

## NEW PAGES ARE IDENTIFIED

by the change number at the bottom of the page. "New" pages are those which were not present when the current edition of the manual was published.

## WRITE-IN CHANGE INSTRUCTIONS

are presented on the following pages of this Manual Change Notice in procedural form.

## TO UPDATE YOUR MANUAL

proceed to the next page of this Manual Change Notice and follow the procedures in sequence.



TECHNICAL MANUAL CHANGES  
(92060-90014)

CHANGE 1:

Reason for Change 1: Incorporate technical manual enhancements.

- A. Replace existing pages with enclosed updated pages.

✓  
yes done  
29/11/78

# PREFACE

This publication is the reference manual for the RTE Interactive Editor (EDITR). EDITR is included as a standard feature of RTE-II, RTE-III, and RTE-IV. EDITR functions are exactly the same under all versions of RTE.

If you want to use EDITR for simple editing tasks, read Sections I and II. The only important thing you need to know about RTE or the File Manager is how you want to start and end your editing session.

If you want to enlarge your knowledge of how EDITR works and wish to make more efficient use of its features, read Sections I, III and IV. These sections assume you are familiar with RTE and the File Manager and that you understand the implications of using Logical Source tracks, working tracks, timed lists, etc.

If you are using EDITR in a Multi-Terminal Monitor or Batch environment, read Section V or VI before you begin editing.

In a multipoint environment (terminals operating under driver DVR07), there are several special considerations for EDITR operation. These are explained in Section VII. *NOT RECORDED*

EDITR is used to create and edit ASCII files. This manual provides a complete description of EDITR commands and functions. Operating system considerations are indicated, and the reader is referred to other RTE system manuals if additional information is required.

This manual is divided into six sections and an appendix.

- Section I — introduces EDITR, describes how to initiate interactive and batch jobs, surveys its operation, describes keyboard conventions and display formats, describes a sequence of decisions to be made about the environment of EDITR, and explains the on-line loading of EDITR.
- Section II — describes three basic functions of EDITR; line edits, character edits and pattern edits. Introduces commands and examples which demonstrate these functions.
- Section III — lists all EDITR messages, with causes and recovery procedures.
- Section IV — explains the use of each command. Commands are grouped by function and examples are provided to show how each is used. Any special features or warnings are also provided.
- Section V — lists the special considerations for using EDITR in a batch environment, and provides instructions for initiation with and without spooling.
- Section VI — defines multi-terminal operation and directs EDITR use in this environment.
- APPENDIX — provides an editing session example showing the commands used in combination.

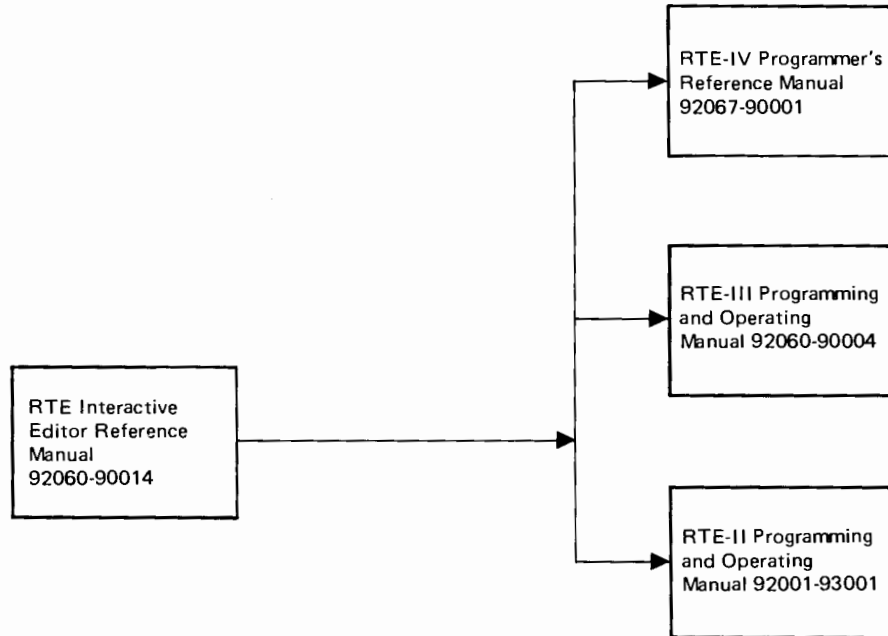
Other publications which should be available for reference when using this manual are:

Batch-Spool Monitor Reference Manual	(92060-90013)
RTE-IV Programming and Operating Manual	(92067-90001)
RTE-III Programming and Operating Manual	(92060-90004)
RTE-II Programming and Operating Manual	(92001-93001)

Additional helpful information can be found in the publications listed on the RTE map.



## DOCUMENTATION MAP



Refer to the documentation map in the appropriate RTE Operating System Reference Manual for a complete listing of relevant documents.

# CONTENTS

Section I	Page	Line Edits	4-17
<b>INTRODUCTION</b>		Replace Pending Line With Text	4-17
EDITR Work Areas	1-1	Insert Text Before Pending Line	4-18
Pending Line	1-2	Insert Text After Pending Line	4-18
Keyboard Conventions	1-2	Delete A Number of Lines	4-19
Display Formats	1-4	Character Edits	4-20
Calling EDITR	1-4	Replace Characters	4-21
Logical Unit	1-4	Insert Characters	4-22
Line Length	1-4	Cancel Characters	4-23
EDITR Prompt Character	1-5	Truncate Characters	4-23
File Definition	1-5	Character Edit Combinations	4-24
Edit Existing File	1-6	Character Edit Summary	4-24
Create File With EDITR	1-6	Pattern Edits	4-24
EDITR Termination	1-7	Search Commands	4-25
Loading EDITR On-Line	1-7	Find a Line With a Find Field-SOF to EOF	4-26
		Find a Line With a Find Field From Pending	
		Line to EOF	4-27
		Delete Lines to Find Field or EOF	4-28
		Jump to Find Field And Make it New	
		Pending Line	4-31
		Exchange Commands	4-33
		Character Replace on Pending Line	4-34
		Exchange on Pending Line, Display Next	
		Occurrence of Pattern	4-35
		Enable Exchange Pattern All Lines(List)	4-36
		Enable Exchange Pattern All Lines(No List)	4-36
		Unconditional Character Replace(List)	4-37
		Unconditional Character Replace(No List)	4-38
		Terminate Commands	4-39
		Abort	4-39
		Assign Edited File to LS Tracks(EL)	4-40
		End Edit and Create a File Manager File(EC)	4-41
		Replace Old File With New File(ER)	4-42
		End Edit, Create File Manager File, Store in	
		LS Tracks(ELC)	4-43
		End Edit, Replace File and Name, and Store	
		in LS Tracks(ELR)	4-44
		Section V	Page
		<b>EDITR IN BATCH ENVIRONMENT</b>	5-1
		Section VI	Page
		<b>EDITR WITH MULTI TERMINAL MONITOR</b>	
		Running EDITR Under RTE-IV Multi-Terminal	
		Monitor	6-1
		Section VII	Page
		<b>EDITR IN MULTIPOINT ENVIRONMENT</b>	
		Introduction	7-1
		Q and O Commands	7-1
		Tab Control in Multipoint Environment	7-2
		<b>APPENDIX</b>	
		Edit Session	A-2
		Compiler Listing of &FLIST	A-5
		<b>INDEX</b>	

# ILLUSTRATIONS

<b>Title</b>	<b>Page</b>
File/Work Area Relationship .....	1-1

# TABLES

<b>Title</b>	<b>Page</b>	<b>Title</b>	<b>Page</b>
Command Syntax .....	1-3	EDTR Message Summary .....	3-1
Keyboard Conventions .....	1-3	EDTR Command Summary .....	4-2
Maximum Line Lengths For Common Devices .....	1-5	Character Edits Compared .....	4-24

The RTE Interactive Editor (EDITR), is commonly used for:

- Creation of new programs or data files in ASCII code.
- Modification of new or existing ASCII files.
- Appending several ASCII files to each other to form a single file.

EDITR operates in either interactive or in batch mode. When used interactively, EDITR accepts operator commands from a keyboard device or a paper tape reader. When used in a batch job, with spooling, EDITR commands are included as part of the job deck. Refer to Section V.

## 1-1. EDITR WORK AREAS

EDITR references a source file and an output file and uses two temporary work areas on disc. The source file is read into a work area on disc called the source work area. Edited text is passed to another work area, also on disc, called the destination work area. An editing pass is completed when EDITR has read and passed all of the text lines in the source work area. Before another editing pass can be made, the source work area is replaced by the destination work area, and the destination work area is cleared.

If you back up to edit a line preceding the current line, the editing pass is completed. The remaining information in the source work area is passed to the destination work area and the "destination work area" now becomes the "source work area."

When editing is complete and EDITR is terminated with one of the terminate commands, the remaining data in the source work area is passed to the destination work area, and the destination work area is written to an output file. The relationship between these files and work areas is shown in Figure 1-1.

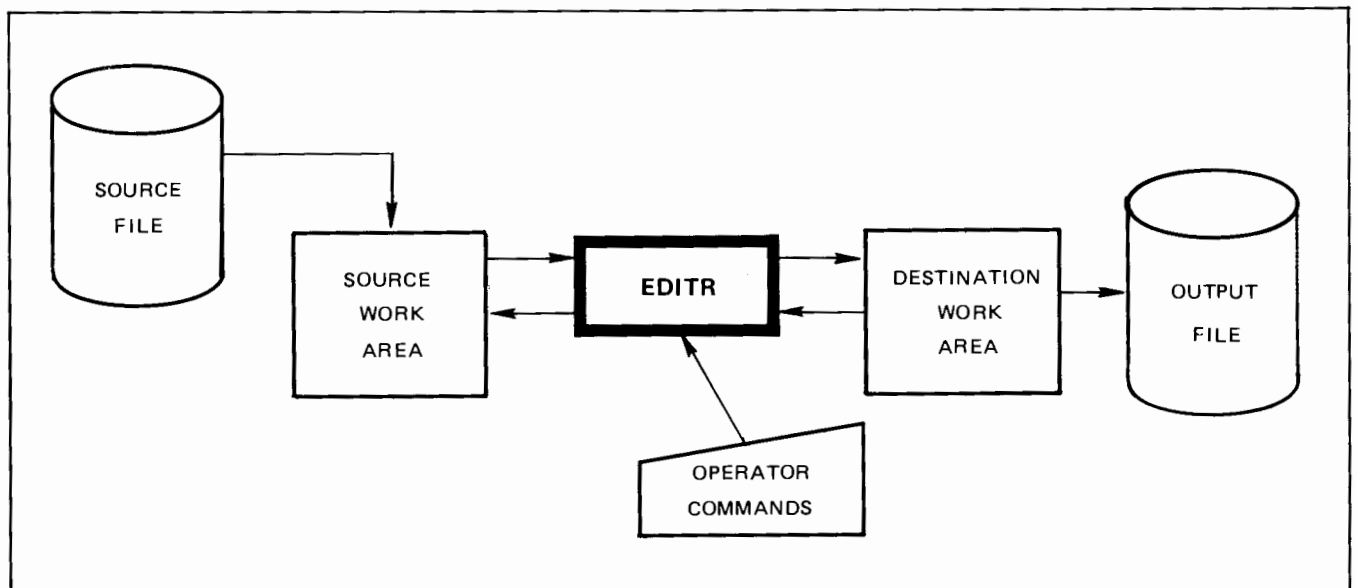


Figure 1-1. File/Work Area Relationship

## 1-2. PENDING LINE

When EDITR is run, and the source file you name at the beginning of the editing session is read into the source work area, the first line is displayed. The displayed line is the current line available for editing and is called the “pending line”. This line remains the pending line until you request a new pending line with an EDITR command. In this way you can continue to re-edit the same line until you are satisfied.

When you request a particular line of text, EDITR searches through the source work area until the requested line is encountered. That line then becomes the new “pending line” and is displayed on the system console. EDITR maintains a “pending line” pointer into the source area.

When the new pending line is displayed, the old pending line, and all other lines passed over by the pointer in the search for the new pending line, are usually written to the destination work area, and are no longer accessible until the “destination work area” becomes the “source work area”. In other words, you cannot go from line 0028 back to line 0024 without the source work area being replaced by the destination work area. Generally you do not have to be concerned with this exchange because it is done automatically. You do need to remember, however, that once the “source work area” is replaced, the previous “source work area” is gone, totally, along with any lines deleted or changed in the last pass of EDITR. You should also remember that if you have made any insertions or deletions, these lines are no longer accessible by their original line numbers. For example, if line 3 is deleted in one editing pass, the original line 4 will be the new line 3. EDITR permits lines of text to be accessed without using line numbers.

## 1-3. KEYBOARD CONVENTIONS

Keyboard commands are used to direct EDITR to do replacements, insertions, deletions, searches and exchanges of text. These functions can be performed on characters within a line; they can be used to manipulate one entire line; or they can be used on groups of lines.

To use EDITR efficiently, you must know the command syntax and keyboard conventions it expects. They are shown and described in Tables 1-1 and 1-2.

### Conventions Used in Examples

△ Space entered at the keyboard.

Ⓧ Circled lower case italic characters indicate a nonprinting control character entered at the keyboard.

Table 1-1. Command Syntax

CONVENTION	MEANING	COMMAND	EXAMPLE	COMMENTS
UPPER CASE BLOCK LETTERS	Literals that must be specified as shown .	F	F	No parameters
<i>lower case italics</i>	Variables to be replaced by values as defined in text .	<i>Wa,b</i>	W7,9	Values 7 and 9 replace variables <i>a</i> and <i>b</i>
<i>UPPER CASE ITALICS</i>	Nonprinting control characters entered by pressing CTRL and a key simultaneously.	<i>CTRL/key</i>	CTRL/C or ⓐ	In examples, control characters are indicated by circled key
[ ]	Bracketed parameters are optional; if omitted default values are supplied.	<i>file name [:security code[:cartridge reference]]</i>		
		Examples: MYFILE:AA:2 URFILE::-13		

Table 1-2. Keyboard Conventions

KEY	MEANING	FORMAT	EXAMPLE	COMMENTS
ESC	Escape key used in pattern edits to define field of indefinite length.	ⓔ or ~	FⓔASMB,R,L,T	Nonprinting character
RETURN	Carriage return used at end of line of input to transmit line to EDITR.	ⓐ	/RNEW LINEⓐ	Nonprinting character
SPACE BAR	1. In place of source file name, causes EDITR to use contents of LS tracks as text. 2. As a command, inserts a line of text after pending line. 3. In text to indicate spaces.	△	SOURCE FILE? /△ /△INSERT LINE /△MIN△△△BSS△1	Nonprinting
RUBOUT	Key used to delete line if pressed before RETURN key enters line; it causes line feed and carriage return to start of next line where correct line can be entered.		/RLINK\ RLINEⓐ	Prints as backslash; prompt is not repeated in next line
BACKSPACE or CTRL/A	Key used to backspace one character; effectively deletes one character each time key is pressed; some terminals use CTRL/H.		/RLINK E ← ⓐ	May print as backspace or underscore

## 1-4. DISPLAY FORMATS

The pending line displayed and listings output by EDITR are always preceded by two blanks. This convention allows room for the EDITR prompt and a single character command, and results in the new text being automatically aligned with that displayed by EDITR. Error messages, in contrast, always begin in column 1. This is so that the difference between an EDITR message and a line of text can be easily seen. For example:

EOF	EDITR text
EOF	EDITR message

## 1-5. CALLING EDITR

You can call EDITR from RTE with either the ON command or the RU command, and from FMGR with the RU command. Several special considerations also apply to the RU command in a Multi-Terminal Monitor environment. The RTE and Batch-Spool Monitor manuals contain the full explanations and implications of both commands.

The two optional parameters passed to EDITR when it is scheduled are:

- lu* — the logical unit number of the device to be used for command input. The default for this parameter is the system console. If you are operating under the Multi-Terminal Monitor option of RTE, this parameter will be set to the *lu* of the terminal being used if not specified.
- line length* — the maximum output line length, in characters; where the default and maximum size is 150 characters. Any line longer than 150 characters, or the specified length, will be truncated.

For example, to schedule EDITR from the File Manager with commands input from the system console and lines limited to 72 characters, the command would be:

```
:RU,EDITR,,72
```

## 1-6. LOGICAL UNIT

The logical unit may specify any device on which you can enter EDITR commands. Usually it is an interactive keyboard device. If you are editing in batch mode as described in Section V, this device could be a read-only device such as a card reader. If you are using batch with spooling, this device could be *lu5* (command input), or an *lu* corresponding to a spool file containing the commands.

## 1-7. LINE LENGTH

You can specify a line length which is compatible with the device your output file is going to. Otherwise, long lines may be truncated. Table 1-3 shows the output devices commonly used with EDITR and the number of characters per line each supports. You should remember that because of the two space convention used in displays, printed lines may be 74 characters.

Table 1-3. Maximum Line Lengths for Common Devices

DEVICE	LINE LENGTHS	COMMENTS
Punched Cards	80 Characters	
CRT	72-80 Characters	
Magnetic Tape	150 Characters	Default maximum length supported by EDITR.
TTY Device	72 Characters	Longer lines will encounter printing problems.
Paper Tape	150 Characters	Default maximum length supported by EDITR.
Disc	150 Characters	Default maximum length supported by EDITR.
Line Printer	80-132 Characters	Number of characters varies with printer model. Consult appropriate manual for your printer.

### 1-8. EDITR PROMPT CHARACTER

When EDITR is used in the interactive mode, it prompts for input with a slash (/). The X command (refer to 4-3) can be used to change this prompt character to any other character. Thereafter, the specified character is output as the EDITR prompt.

### 1-9. FILE DEFINITION

The File Manager parameter *NAMR* is defined as a file name followed by two subparameters. The subparameters may be omitted from the end of the list. If an embedded subparameter is omitted, its position must be indicated by the colon (:).

*file name[:security code[:cartridge reference]]*

*file name*

1-6 ASCII characters identify file.

- must be printable characters
- first character must not be blank or integer
- embedded blanks are not allowed
- + - , : are not allowed.

*security code*

integer or two ASCII characters identifying security code assigned to file

= 0 not protected (default)

> 0 ~~read~~ <sup>WRITE</sup> protected

< 0 write/read protected.

*cartridge reference*

label or logical unit identifying cartridge where file resides

= 0 unspecified cartridge (default)

> 0 integer or cartridge label; ASCII identifier

< 0 logical unit of cartridge, i.e., -14 is logical unit 14.



## Introduction

### 1-10. EDIT EXISTING FILE

When EDITR begins execution, it requests information about the file to be edited and prompts the user:

SOURCE FILE?

/

At this point, there are four legal responses you can make, each of which must be followed by a carriage return:

- 1) 0 (zero)
- 2) : (colon)
- 3) *filename*
- 4) Δ (blank)

If you specify "0", you will automatically begin working with an empty file. The EDITR commands can then be used to enter and edit your lines of text. See paragraph 1-11 for an illustrative example.

If you specify ":", the EDITR is immediately aborted and control returns to the program from which the EDITR was scheduled.

If you specify *filename*, the contents of the file will be copied into the EDITR's source work area. The number of characters in *filename* cannot be greater than the current line length.

The file may be a type 0 file to read source information directly from a peripheral device. *Filename* may be specified with or without a security code and a cartridge label. For example, to specify a file without a security code, but with a cartridge label:

/FILE1::27

Refer to the discussion on type 0 files and the *namr* parameter in the BATCH-SPOOL Monitor Reference Manual for more information.

If you specify "Δ", the current Logical Source (LS) area of the disc is copied into the EDITR's source work area. The EDITR commands may now be used to edit the source text.

### 1-11. CREATE FILE WITH EDITR

To create a file with the EDITR, enter a 0 (zero) in response to the EDITR prompt "SOURCE FILE?". EOF is printed and you can proceed to enter the lines for your file as illustrated:

```
:RU,EDITR
SOURCE FILE?
10 ←————— Enter 0 to put empty file into EDITR'S source work area
EOF
/ΔNEW FILE ←————— Enter a space (for insert) and then line of text
/ΔLINE TWO ←—————
/ΔLINE THREE ←—————
/ECFILE1 ←————— End EDITR and Create FMGR file named FILE1
```

The above command sequence will enter the three lines shown into the file name FILE1.

## 1-12. EDITR TERMINATION

The EDITR terminate commands assign the final version of text in the destination area to the RTE system Logical Source (LS) tracks, and/or to a File Manager type 4 file. The destination area can be output directly to a device through a type 0 file.



## 1-13. LOADING EDITR ON-LINE

The EDITR program can be loaded on-line using the RTE Relocating Loader (LOADR). The minimum recommended size of the program is six pages, with seven being suggested. EDITR uses the extra space for its source and destination work areas. The following example presents a typical on-line load of the EDITR.

```
*RU,LOADR
  /LOADR:  SZ,7
  /LOADR:  RE,%EDITR
  /LOADR:  EN
```

The EDITR should be specified as a Type 2 or Type 3 program when it is loaded on-line or during generation.



# EDITR FUNCTIONS

SECTION

II

You can use EDITR to access and manipulate text in the source work area in three ways:

- Line edits  
which insert, delete, or replace one line of text at a time.
- Character edits  
which insert, delete, or replace characters within the current line
- Pattern edits  
which automatically insert, delete, or replace character strings over many lines of text in the source work area

All edits operate on the pending line. The pointer that moves through the source file from line to line and defines the pending line will also, for character edits, point to a particular character within the line. Pattern edits cause this pointer to move automatically through the source file searching for a particular line or group of lines to edit. As each line is found, it becomes the new pending line and can be edited. In all cases, as the source work area is edited, the results are placed in the destination work area.

The source work area is replaced by the destination work area whenever one of the following occur:

- a) the source pointer is moved to a line prior to the current line
- b) one of several commands explained in Section IV are used.

## 2-1. LINE EDITS

You can use line edits to insert, delete, or replace one complete line of text. The commands are R to replace a line, I to insert before a line, Δ (space) to insert after a line, and - (minus) to delete a line.

To replace a line of text, use R. For example:

```
/1 ← Display line 1.  
ASMB,R,L,T  
/RASMB,R,B,L ← Enter new line following R.  
/P  
ASMB,R,B,L ← Display edited line.
```

To insert a line of text before the pending line, use I. For example:

```

/4 ← EXT .ENTR, EXEC
/I DEF TBUF ←
/3 ← ENT WEN
/L3 ← ENT WEN
      DEF TBUF
      EXT .ENTR, EXEC
      HRS BSS 1
  
```

Make line 4 the pending line.  
 Enter the new line following the I.  
 Request a line number smaller than the current line number to force an exchange between source work area and the destination work area.  
 List three lines, beginning with the new pending line number, and note the new line.

To insert a line of text after the pending line, use a space (Δ). For example:

```

/6 ← HRS BSS 1
/Δ MSC BSS 1 ←
/5 ← EXT .ENTR, EXEC
/L3 ← EXT .ENTR, EXEC
      HRS BSS 1
      MSC BSS 1
      MIN BSS 1
  
```

Make line 6 the pending line.  
 Enter the new line following the space.  
 Request a line number smaller than the current line number to force the destination work area to replace the source work area.  
 List three lines of the new source work area after the pending line to show that the line has been inserted.

To delete one line of text use a minus (-). This deletes the pending line. For example:

```

/L3
  ASMB, R, L, T
      NAM TIME
      ENT WEN
      EXT .ENTR, EXEC
/3 ← ENT WEN
/- ← EXT .ENTR, EXEC
/1 ← ASMB, R, L, T
/L3
  ASMB, R, L, T
      NAM TIME
      EXT .ENTR, EXEC
      HRS BSS 1
  
```

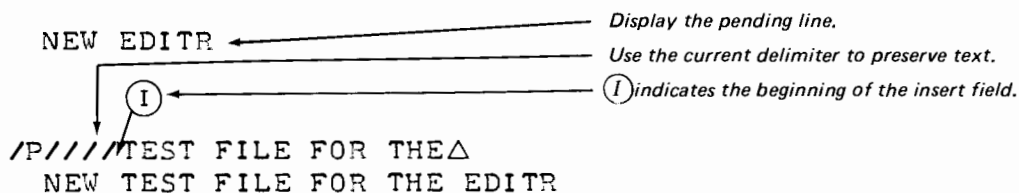
List three lines of file so that the deletion can be verified.  
 Position so that the line to be deleted is the pending line.  
 Enter the command to delete. EDITR responds with the new pending line.  
 Return to line 1 and list three lines as before.  
 The line ENT WEN is no longer in the source work area.

## 2-2. CHARACTER EDITS

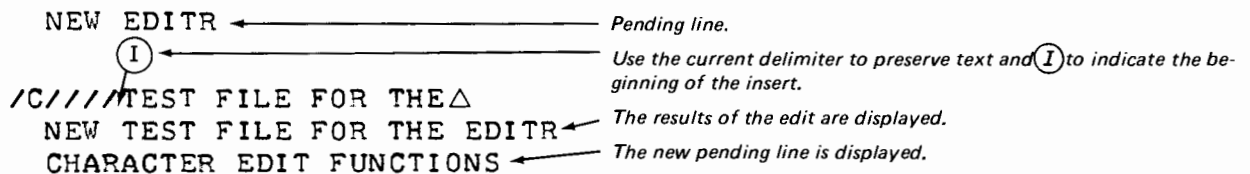
You can change characters in the current line with character edits. All character edits are made to the pending line. Either "P" or "C" must be the first character entered. If the control character is "P", the altered line remains the pending line and the results of the edit are displayed. If "C" is the first character entered: the results of the edit are displayed, the altered line is passed to the destination work area, and the next line of the source work area is displayed as the new pending line.

The current delimiter is used to space over characters that are not to be changed. Various control characters: **I** or **S** for insert, **T** for truncate, **C** to cancel, and **R** for replace, are used to further define the basic character edit commands C and P. Control characters do not apply in a multipoint environment; refer to Section VII for character edits in a multipoint environment.

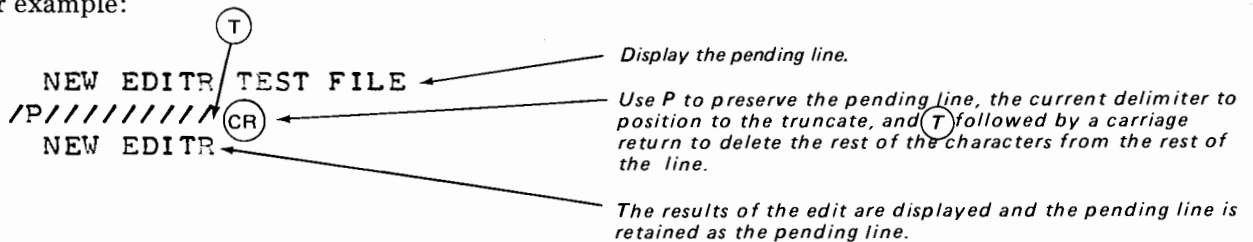
To insert characters on the pending line and keep the altered line as the pending line, use P followed by **I**. For example:



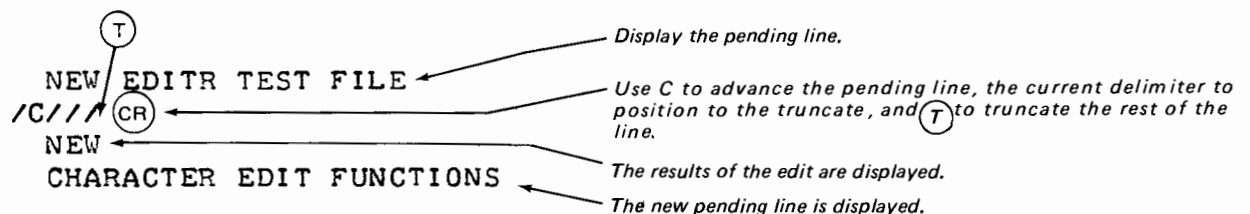
To insert characters on the pending line and automatically advance to the next line, use C followed by **I**. For example:



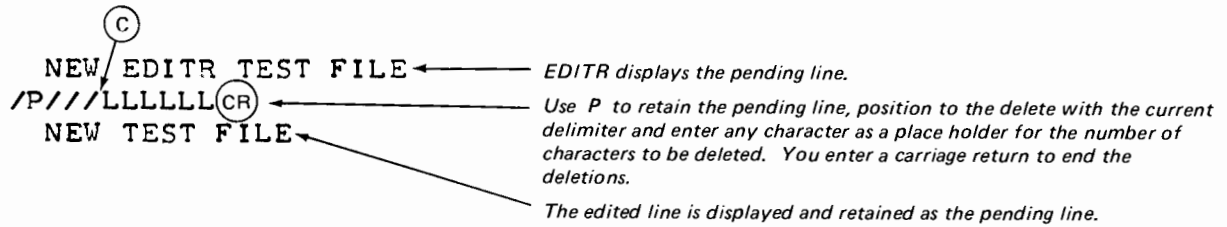
To truncate characters from the pending line and keep the altered line as the pending line, use P followed by **T**. For example:



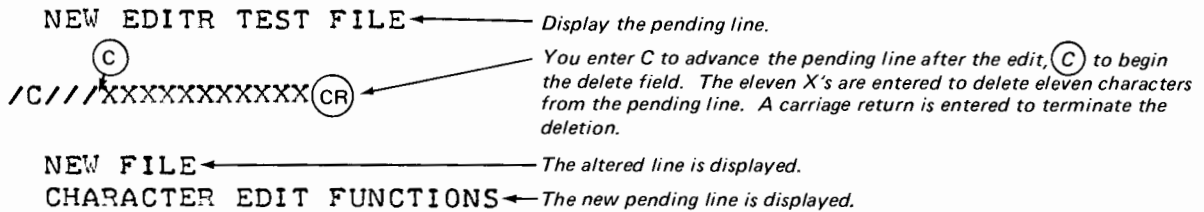
To truncate characters from the end of the pending line and automatically advance to the next line, use C followed by **T**. For example:



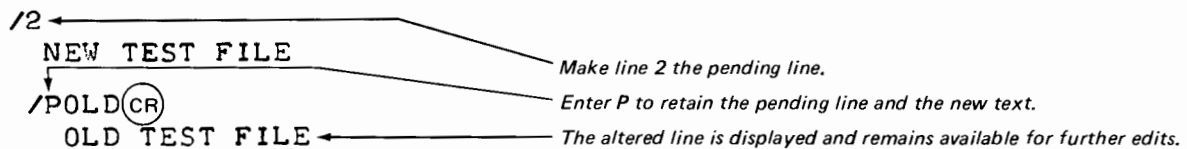
To delete characters from within the pending line and keep the altered line as the pending line, use P, followed by **(C)**. For example:



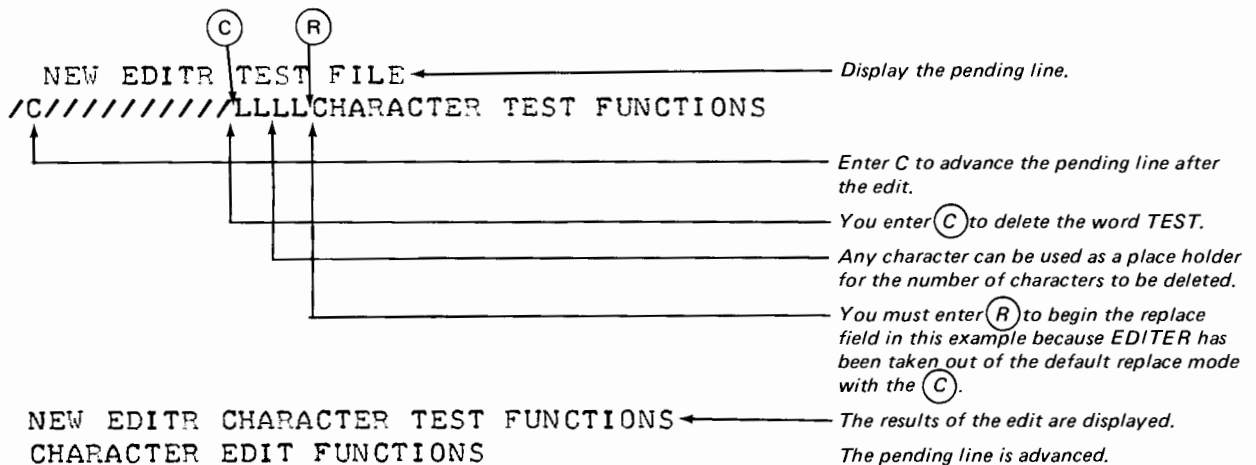
To delete characters from within the pending line and automatically advance to the next line, use C followed by **(C)**. For example:



To replace characters in the pending line if no other control characters have been entered, use P and the new text you wish to enter. The current delimiter is still used to preserve text so that you can position anywhere in the line.



To replace characters in the pending line if other control characters have already been entered, **(R)** must be entered to define the replace field, in addition to either P or C.

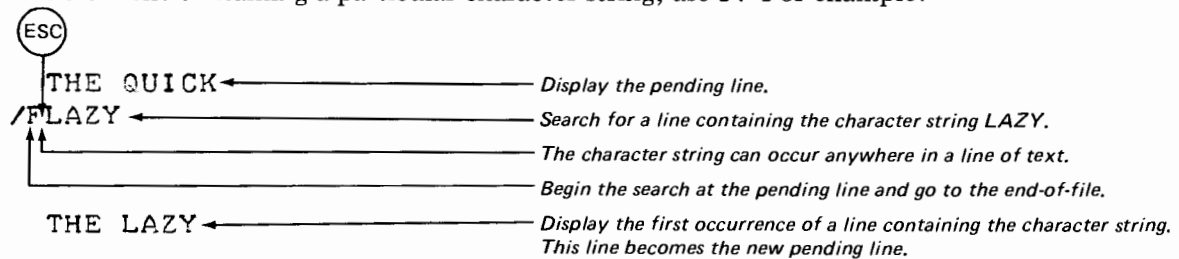


## 2-3. PATTERN EDITS

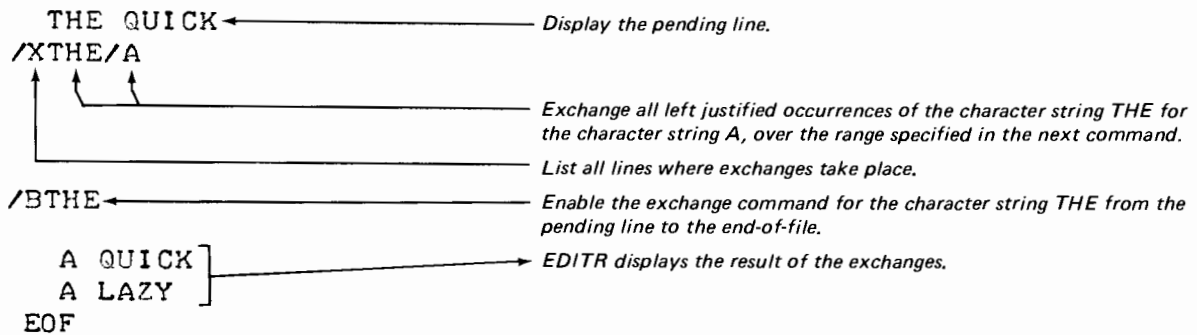
Pattern edits are used to look for a specific character string and to replace that string with another whenever the pattern is recognized. There are two kinds of pattern edits: searches and exchanges. A search matches a string of characters entered at the keyboard with a corresponding string in the text. To perform an exchange, you enter two sets of character strings at the keyboard. Whenever the first string is encountered, it is replaced by the second string.

The control characters you specify for the various search commands determine the initial pointer positions and the length of the block of data involved. Search and exchange functions can also be combined. Selective exchanges can be done, for instance, by first doing successive searches and then exchanges on lines that meet some criteria, i.e., every other occurrence of the pattern, in addition to the search pattern.

To find a line of text containing a particular character string, use F. For example:



To exchange one character string for another from the pending line to the end-of-file, use the X and B commands. For example:







# EDITR MESSAGES

SECTION

III

Errors encountered by the Editor and information messages are reported on the operator console. A summary of the messages, their effects, and error recovery procedures are contained in Table 3-1.

Table 3-1. EDITR Message Summary

MESSAGE	DESCRIPTION	SUGGESTED ACTION
EDITR ABORTED	If this message is displayed after you press "A", to abort EDITR, there is no error. Otherwise, the RTE or Batch-Spool Monitor systems have aborted EDITR. The source file remains unchanged. The destination file is deleted.	Rerun EDITR
??	Syntax error in the command just given to EDITR, or the input device has timed out waiting for a command.	Retype the command
EOF	A command has caused an attempt to read beyond the current end of the source file.	Return to beginning of file by typing in a 0 or 1.
CORRUPT FILE	File has a record of more than 150 characters. Usually caused by LS pointing at something besides a file.	Check LS pointer. Rerun EDITR using back up copy of source.



# EDITR COMMANDS

SECTION

IV

The description of the Editor commands follow, as closely as possible, the format displayed at the console. A slash (/) is used as the current delimiter to request user input.

## 4-1. PROGRAM LISTING

A complete listing of the program (TIME), used in the command examples is provided here for reference. The example was produced by using the FMGR :LI command so that the file name and line numbers can be used for correlation.

```
TIME    T=00004 IS ON CR00002 USING 00003 BLKS P=0024

0001  ASMB,R,L,T
0002          NAM TIME
0003          ENT WEN
0004          EXT ,ENTR,EXEC
0005  HRS     BSS 1
0006  MIN     BSS 1
0007  SEC     BSS 1
0008  WEN     NOP
0009          JSB ,ENTR
0010          DEF HRS
0011          JSB EXEC
0012          DEF **3
0013          DEF TCODE
0014          DEF TBUFF
0015          LDA TBUFF+3
0016          STA HRS,I
0017          LDA TBUFF+2
0018          STA MIN,I
0019          LDA TBUFF+1
0020          STA SEC,I
0021          JMP WEN,I
0022  TCODE   DEC 11
0023  TBUFF   BSS 5
0024          END
```

## 4-2. EDITR COMMANDS

All of the EDITR commands are summarized in Table 4-1. This table presents the commands in the same functional groups in which they are described in this section.

Table 4-1. EDITR Command Summary

FUNCTION	COMMAND	DESCRIPTION	PAGE NO.
CONTROL	X	Change EDITR Prompt Character	4-4
	CTRL/G	Bell Control	4-4
	;	Set Tab Stops	4-5
	W	Set Window	4-5
	#	Sequence Numbers	4-6
	=	Set Line Length	4-7
	K	Kill Trailing Blanks	4-8
	<i>Mnamr</i>	Merge Source File Following Pending Line	4-9
DISPLAY	P	Display Pending Line	4-11
	<i>L</i> <i>n</i>	Display a Number of Lines	4-11
	<i>n</i>	Display Specified Line	4-12
	/ or +	Space Down a Number of Lines and Display	4-13
	N	Display Pending Line Number	4-14
	ND	Display the Number of Lines in the Destination Work Area	4-14
	H	Display the Number of Characters in the Pending Line	4-15
	↑	Back Up in Destination Work Area	4-16
	S	Display Approximate Number of Words in the Destination Work Area	4-16
LINE EDITS	R	Replace Pending Line With Text	4-17
	I	Insert Text Before Pending Line	4-18
	Δ	Insert Text After Pending Line	4-18
	-	Delete a Number of Lines	4-19
CHARACTER EDITS	CTRL/R	Replace Characters	4-21
	CTRL/I or CTRL/S	Insert Characters (CTRL/S Alternate for CTRL/I)	4-22
	CTRL/S	Alternate for CTRL/I)	4-22
	CTRL/C	Cancel Characters	4-23
	CTRL/T	Truncate Remainder of Pending Line	4-23

Table 4-1. EDITR Command Summary (Continued)

FUNCTION	COMMAND	DESCRIPTION	PAGE NO.
PATTERN EDITS	Search Commands		
	;	Find Field Tabbed	4-25
	<i>esc</i>	Find Field of Indefinite Length	4-25
	/	Find Field Within Window	4-25
	CTRL@	Find a Zero Length Line With This Field	4-25
	B	Find a Line With Find Field-SOF to EOF	4-26
	F	Find a Line With a Find Field From Pending Line to EOF	4-27
	D	Delete Lines to Find Field or EOF	4-28
	J	Jump to Find Field Line and Make it New Pending Line	4-31
	Exchange Commands		
	G	Character Replace on Pending Line	4-34
	Y	Exchange on Pending Line, Display Next Occurrence of Pattern	4-35
	X	Enable Exchange Pattern All Lines (list)	4-36
	Z	Enable Exchange Pattern for Next Edit (no list)	4-36
	V	Unconditional Exchange (list)	4-37
U	Unconditional Exchange (no list)	4-37	
TERMINATE	A	Abort EDITR	4-39
	EL	Assign Edited File to LS Tracks	4-40
	<i>ECname</i>	Create File Manager File	4-41
	ER	Replace Old File With New File	4-42
	<i>ERname</i>		
	<i>ELCname</i>	Create File Manager File and Store in LS	4-43
	ELR	Replace Old File With New File and Store in LS Tracks	4-44
	<i>ELRname</i>		

Example:

`/W7,9` ← *In this example, the window is set to include only columns 7, 8, and 9.*

4-8. SEQUENCE NUMBERS

You can use this command to cause EDITR to put sequence numbers on all lines in the file. A three character identifier begins in column 73 and the numbers begin in column 76.

Format:

<code>#</code>	Start numbers in column 76.
<code>#xxx</code>	Start numbers in column 76 preceded by <i>xxx</i> .
<code>#xxx numb1</code>	Start numbers with <i>numb1</i> value. <i>xxx</i> field required.
<code>#xxx numb1, numb2</code>	Start numbers with <i>numb1</i> value, increment by <i>numb2</i> . <i>xxx</i> field required.
Where:	
<i>xxx</i>	is the three character identifier. It occupies columns 73-75 and must be included when specifying <i>numb1</i> and <i>numb2</i> (it may contain blanks, however).
<i>numb1</i>	is the starting number. The first line will start with this number. If omitted, numbers start with 00000.
<i>numb2</i>	is the incrementing value. <i>numb1</i> is incremented by <i>numb2</i> for each line. If omitted, <i>numb1</i> is incremented by 10.

When all parameters are omitted, sequence numbers start in column 76 and a blank identifier is used. Remember that if you provide values for either *numb1* or *numb2*, you must also provide a value for the three character identifiers.

Example:

1. Use default values for command:

```
SOURCE FILE?
/TIME
  ASMB,R,L,T
/#
EOF
```

*All parameters defaulted, results in:*

1. Sequence numbers begin in column 76
2. Numbers consist of 5 digits
3. Numbers are incremented by 10
4. Some digits are lost on a line printer listing.

```
0001  ASMB,R,L,T           00000
0002      NAM TIME       00010
0003      ENT WEN       00020
0004      EXT ,ENTR,EXEC 00030
```

2. Specify all parameters:

```
ASMB,R,L,T ← Incrementing value
/#0011,1 ← Starting number
EOF ← Character field
```

To delete the blanks in columns 60-70 so that the entire sequence number field can be output on the line printer, these commands are entered at the TTY:

```
/0
ASMB,R,L,T
/W60,70 ← Set window to delete blanks.
/Z ← 10 blank spaces.
/FXX
EOF
```

ASMB,R,L,T	00100001
NAM TIME	00100002
ENT WEN	00100003
EXT ,ENTR,EXEC	00100004
HRS BSS 1	
JMP WEN,I	00100021
TCODE DEC 11	00100022
TBUFF BSS 5	00100023
END	00100024

#### 4-9. SET LINE LENGTH

The line length initially defaults to 150 characters when the Editor is turned on. This command allows the user to reset the line length to another value and to truncate any characters in the line beyond the new limit.

Format:

=n

Example:

```
/5 ← Request line number 5.
HRS BSS 1
/=5 ← Set the line length to 5 characters.
/P ← Display the pending line to show the results of the edit.
HRS△△
```

You should remember that if the line length is set to a value less than the number of characters in the name of a File Manager file to be edited, EDITR will not be able to access the file.



```

/1
:LG,10
/F:MR,ST6
:MR,ST6
/MQQ
:RU,LOADR,99,6,28,1,2

```

*The pending line of Q is positioned for the merge with an F command.*  
*The Merge Command names the file to be merged into the source file.*  
*EDITR responds after the merge by displaying the new pending line of the source file.*

Source file Q relisted after the merge.

```

/1
:LG,10
/L500
:LG,10
:MR,STM
:MR,ST1
:MR,MNEMFR
:MR,ST2
:MR,ST3
:MR,ST4
:MR,BRTBFR
:MR,ST5
:MR,ST6
:MR,ST7
:MR,ST8
:RU,LOADR,99,6,28,1,2
:RU,BASIC,1,1,1
:TR
EOF
/ERQ::-2
END OF EDIT

```

*The new listing shows the merged statements.*  
*The edited file in the destination work area replaces the old File Manager file "Q".*

## 4-12. DISPLAY COMMANDS

These commands cause the contents of the source work area, or information about the contents to be displayed. The display normally occurs on the console, but some commands allow displays on the line printer, your terminal, etc.

### 4-13. DISPLAY THE PENDING LINE

This command displays the pending line on the system console. You may then modify it if you wish. The P command can also be used in character edits as described in 4-27.

Format:

P
---

Example:

```
/P
  ASMB,R,L,T
```

### 4-14. DISPLAY A NUMBER OF LINES

This command displays the pending line plus a specified number of lines on a specified device. The list device must be an output device such as a terminal, line printer, punch, etc. If it is not an output device, EDITOR will continue to run and the source area pointer will be advanced, but no lines are displayed. It should be noted that two blanks are inserted as the first two characters of each line displayed and that the new pending line displayed on the system console is the next line after those listed.

Format:

<i>Ln</i>	List <i>n</i> lines
<i>Ln, lu</i>	List <i>n</i> lines on logical unit <i>lu</i>
Where:	
<i>n</i>	is the number of lines to be printed
<i>lu</i>	is the optional logical unit number of the list device. The default is the system console.

## EDITR Commands

### Example:

```
/1
  ASMB,R,L,T
/L5,6 ← Logical unit number specifies the line printer.
  MIN   BSS 1

ASMB,R,L,T
      NAM TIME
      ENT WEN
      EXT .ENTR,EXEC
HRS   BSS 1 ] → Line printer listing.

ASMB,R,L,T
      NAM TIME
      ENT WEN
      EXT .FNTR,EXEC
HRS   BSS 1 ] → List to user console.
```

### 4-15. DISPLAY A SPECIFIED LINE

This command displays the requested line and makes it the pending line. If the line number is less than or equal to the current pending line number, the destination work area replaces the source work area, changing text line numbers, if any insertions or deletions were made to the text.

#### Format:

*n*

#### Example:

```
/12
      DEF **3 ← Initial line numbers.
/13
      DEF TCODE ← Initial line numbers.
/14,6
      DEF TBUFF ← Initial line numbers.
/12
      DEF **3 ← Reposition to insert.
/
      DEF *5
/
      DEF **2 ] ← Inserted lines.
/12
      DEF **3
/13
      DEF *5 ← Line contents changed by insertions.
/14
      DEF **2
/15
      DEF TCODE
```

In the example, it is assumed the first line number is one, the second line number is two, etc. A zero, is interpreted as one.

## 4-16. SPACE DOWN A NUMBER OF LINES

This command advances the pending line the specified number and displays the new pending line.

## Format:

/	or +	Advances the pending line 1.
+n	or /n	Advances the pending line the number specified.
+n, lu	/n, lu	Used after a change command. Advances the number of lines specified. Displays the new pending line at the console and displays all changed lines at the <i>lu</i> specified.

## Where:

*n* is the number of lines to be skipped. The default is 1.

*lu* defines the logical unit number of the device on which the display occurs. The default is the system console.

## Example:

```
ASMB,R,L,T
/+5
MIN BSS 1
/1
ASMB,R,L,T
//5
MIN BSS 1
/1
ASMB,R,L,T
```

## DEFAULTS

```
ASMB,R,L,T
/+
NAM TIME
/1\
1
ASMB,R,L,T
//
NAM TIME
```

To skip over a line of text use +2 or /2 with the results:

```
LINE 1 — Pending line when +2 entered
LINE 2 — Skipped
LINE 3 — New pending line
```

#### 4-17. DISPLAY PENDING LINE NUMBER

This command displays the number of the pending line in the source work area.

Format:

N
---

Example:

```
/1
  ASMB,R,L,T
/+3
      EXT .ENTR,EXEC
/N
  4
```

#### 4-18. DISPLAY LINE NUMBER IN DESTINATION WORK AREA

This command displays the line number of the last line written to the destination work area. The source work area pending line and EOF's are not included in the count.

Format:

ND
----

Example:

```
/1
  ASMB,R,L,T
/ND
/2
  NAM TIME ← Source work area pending line.
/ND
  1 ← Line 2 is not included in count
      because it hasn't yet been transferred to the destination work area.
/25 ← Source work area line number.
EOF
/ND
  24 ← Destination work area count.
```

4-19. DISPLAY NUMBER OF CHARACTERS IN PENDING LINE

This command displays the number of characters in the pending line. The count includes a padded blank if the number of characters is uneven — since EDITR generates word-length records of two characters per word. This padding may be lost during character edits but replaced as the line is added to the destination work area.

Format:

H

Example:

```

/7
SEC△△△BSS△I△
/H
12
/
P
SEC BSS 1
/RSEC△△△BSS△I
/H
11

```

*Padded blank at end of line.*  
*Twelve characters include padded blank.*  
*Character-by-character replacement of line in 1.*  
*Character count does not show padded blank.*

The first time H is used, the number of characters in the line is twelve. After a character-by-character replacement of the line, the number of characters is 11. This is because the padded blank is not inserted until the line is written to the destination work area.

4-20. DISPLAY HEADER

This command displays a header which facilitates column location.

Format:

HL

Example:

```

/HL
'1'/'2'/'3'/'4'/'5'/'6'/'7'/'8'
/

```

#### 4-21. BACK UP IN DESTINATION WORK AREA

This command allows you to erase lines in the destination work area. It does this by causing the pointer in the destination work area to back up. An up arrow (↑) by itself backs up one line. It should be noted that the up arrow command forces a transfer from the source work area to the destination work area.

Format:

```
↑ (n)
```

Example:

```

/5
  HRS   BSS 1
/ND
 4
/↑2
      ENT WEN
/ND
 2
    
```

← Display current line number in destination work area. Back up two lines in destination work area.  
← Display new current line in destination work area.  
← New current line number in destination work area as a result of back up.

If *n* is so large that it causes the pointer to back up past the beginning of the destination work area, the error message ?? is displayed and a smaller value should be specified.

#### 4-22. DISPLAY APPROXIMATE NUMBER OF WORDS IN DESTINATION WORK AREA

The S command prints the approximate number of words in the destination work area (to the pending line -1).

Format:

```
S
```

Example:

```

/N
 24
/ND
 23
/S
 206
/Ø
  ASMB,R,L,T
/K
 EOF
/S
 197
    
```

← Pending line number in source work area.  
← Number of lines in destination work area.  
← Number of words in destination work area, minus pending line.  
← EDITR interprets a 0 as requesting line 1.  
← Destination work area replaces source work area.  
← Delete Trailing blanks.  
← Number of words remaining in destination work area now includes pending line omitted in the first S response, minus trailing blanks.

This command can be used to determine the need to break up files which would result in large paper tapes. A paper tape containing more than 14,000 words exceeds the standard box size. If the word count is determined with the S command, the file can be broken with a series of zero-length records in the appropriate spots.

### 4-23. LINE EDITS

You can manipulate one entire line of text by doing line edits. EDITR provides four commands (R, I, Δ, and 0) to replace, insert and duplicate a line of text, plus one command to delete a line of text (-n).

### 4-24. REPLACE PENDING LINE WITH TEXT

This command replaces the pending line with new text entered at the console. If this command is used with no new text input, the pending line becomes zero length.

Format:



*Rtext*

Example:

```

/6
MIN   BSS 1
/RMAX BSS 5
/6
MAX   BSS 5
    
```

← *Make line 6 the pending line, and display it.*  
 ← *Replace with this new line.*  
 ← *Display new line 6.*



## EDITR Commands

### 4-25. INSERT TEXT BEFORE PENDING LINE

The I command inserts a new line of text before the pending line. If this command is entered without new text, the inserted line becomes zero length.

Format:

*Itext*

Example:

```
/L5
  ASMB,R,L,T
      NAM TIME
      ENT WEN
      EXT .ENTR,EXEC
HRS   BSS 1
MIN   BSS 1

/5 ←
HRS   BSS 1
/I SHR BSS 1 ←
/L5
  ASMB,R,L,T ←
      NAM TIME
      ENT WEN
      EXT .ENTR,EXEC
SHR   BSS 1 ←
HRS   BSS 1
```

*Move pointer to line 5.*  
*Insert line of text.*  
*Move pointer to beginning of file and display five lines of new text.*  
*Inserted line.*

### 4-26. INSERT TEXT AFTER PENDING LINE

This command, which is indicated by a space, is used to insert new text immediately after the pending line. If this command is used without new text, the new line has zero length. The new line becomes the pending line.

Format:

*Δtext*

Example:

```
/4      EXT .ENTR,EXEC
/ HHR   BSS 1
/4
      EXT .ENTR,EXEC
/L2
      EXT .ENTR,EXEC
HHR   BSS 1
HRS   BSS 1
```

*Make line 4 the pending line.*  
*Insert a line of text after pending line.*  
*Go back to line 4 to verify insertion.*  
*List line 2 and next two lines to verify insertion follows pending line.*

4-27. DELETE A NUMBER OF LINES

This command deletes a specified number of lines in the text. If no value is specified, one line is deleted.

Format:

```
-n
```

Example:

```
TIME1 T=00004 IS ON CR00002 USING 00002 BLKS R=0010
```

```
0001 ASMR,R,L,T
0002     NAM TIME
0003     ENT WEN
0004     EXT .ENTR,EXEC
0005 HHR     BSS 1
0006 SRH     BSS 0
0007 IRS     BSS 3
0008 MAX     BSS 5
0009 ISC     BSS 7
0010 NEW     JSB .ENTR
```

*File Manager listing of source using :L1 command to show line numbers.*

```
/4 ← EXT .ENTR,EXEC
/-5 ← JSB .ENTR
```

*Position to line number 4.*

*Delete 5 lines including line number 4.*

*Display new pending line.*

```
TIME1 T=00004 IS ON CR00002 USING 00002 BLKS R=0010
```

```
0001 ASMB,R,L,T
0002     NAM TIME
0003     ENT WFN
0004 ISC     BSS 7
0005 NEW     JSB .ENTR
```

*File Manager listing to show results of delete.*

## 4-28. CHARACTER EDITS

The character edit commands provide a means to change the contents and modify the structure of the current line of text. Four commands allow the replacement, insertion, and deletion of characters. Each of these commands uses nonprinting control characters so that character alignment is maintained in the text. Each must be used with an initial “P”, “C”, or “O” command.

The current delimiter is used as a “place holder” to preserve existing text in the pending line. The P, C and O commands are not themselves character edit functions. They are used to determine the line disposition during the edit: P leaves the edited line as the pending line, C advances the pending line to the next line after the edit, O sends the pending line to the destination work area and then edits a copy of the line and leaves it pending.

Special considerations apply to these commands in a multipoint environment. Refer to Section VII for a complete explanation on how to use the EDITR in that environment.

When C is entered as the first character:

- the pending line is edited
- the results of the edit are displayed
- the edited line is passed to the destination work area
- the next line of code is displayed and is the new pending line.

When P is entered as the first character:

- the pending line is edited
- the results of the edit are displayed
- the edited line remains the pending line, allowing you to make further edits to this line.

When O is entered as the first character:

- the pending line is passed to the destination work area
- a copy of the pending line is retained and is edited
- the results of the edit are displayed
- the edited line is retained as the pending line.

4-29. REPLACE CHARACTERS

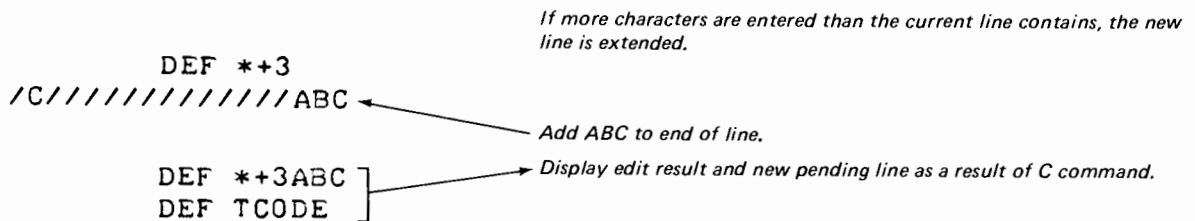
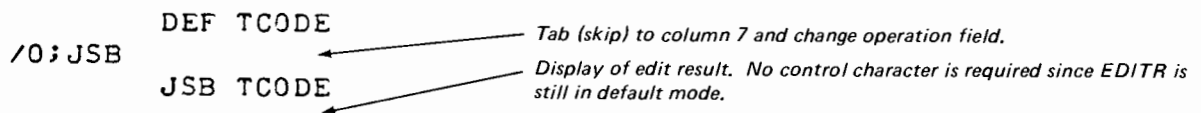
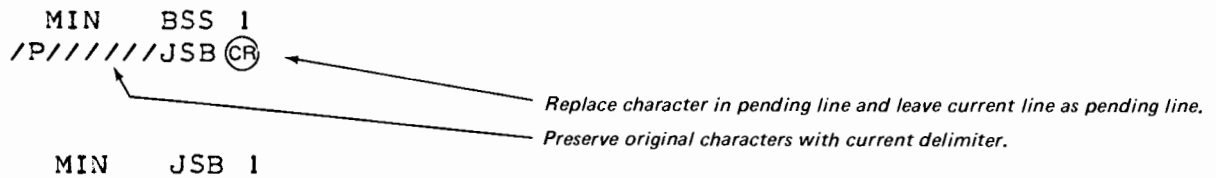
Character replacement allows you to change any character in the pending line. Any embedded character you wish to preserve in the original text is skipped by entering the current delimiter. Using the tab character will also cause characters to be skipped. Skipped characters in the new line appear exactly as they did in the old line. A carriage return preserves the rest of the line unless CTRL/T was specified to truncate.

Format:

P CTRL/R	(Refer to Section VII for multipoint operation).
C CTRL/R	
O CTRL/R	

Character replacement is the EDITR default mode so that the control character CTRL/R can be omitted when EDITR is initially turned on, or following a P, C, or O command. It is only required following one of the control characters CTRL/I, CTRL/S, or CTRL/C.

Example:



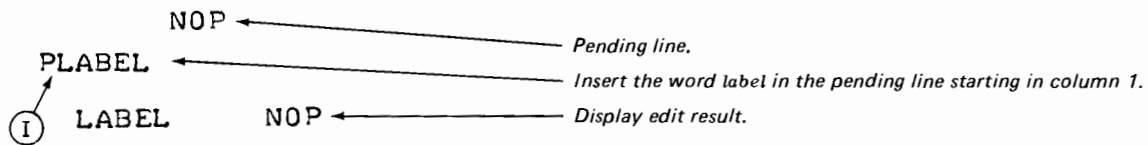
4-30. INSERT CHARACTERS

This command is used to insert characters in the pending line. There are two formats for this command: CTRL/I and CTRL/S. CTRL/I is the standard format. CTRL/S is used on terminals where CTRL/I has a special function (e.g., the HP 2754 (ASR35) teleprinter uses CTRL/I as a tab function). Each new character is inserted in the line immediately before the character under which the CTRL/I or S is entered. Once the control character is entered, entering the current delimiter to do character skipping inserts blanks. The tab character also causes the tabbed number of blanks to be inserted.

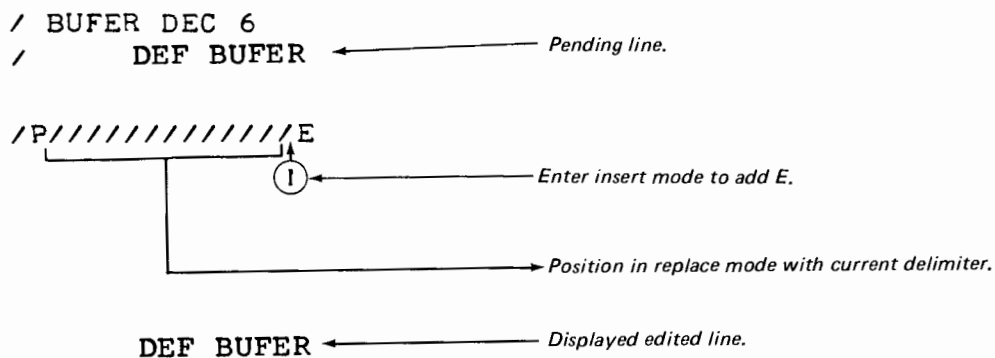
Format:

P CTRL/I		P CTRL/S	(Refer to Section VII for multipoint operation).
C CTRL/I	or	C CTRL/S	
O CTRL/I		O CTRL/S	

Example:



Note that CTRL/I or S is a nonprinting character and appears in the example with a circle around it for clarity. Since characters cannot be skipped in the Insert Mode, all skipping must be done first, in the Replace Mode, before changing the mode to Insert. To insert a letter in a word, as in the example below, the position is aligned in the Replace Mode, and then the mode is changed to Insert.



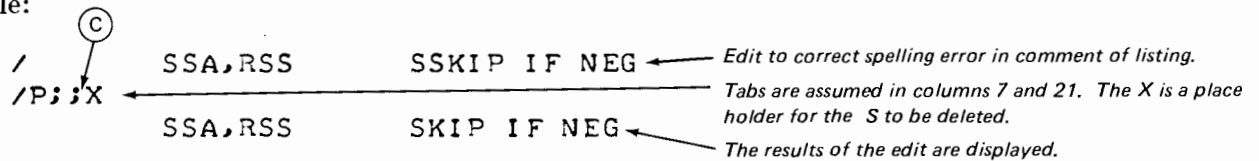
4-31. CANCEL CHARACTERS

This command is used to delete characters from the pending line. Each character or place holder entered after CTRL/C will delete one character from the pending line. Character skipping, i.e., entering the current delimiter deletes characters. The tab character deletes everything up to the tab stop. When carriage return is entered, the pending line will be left justified.

Format:

```
PCTRL/C (Refer to Section VII for multipoint operation).
CCTRL/C
OCTRL/C
```

Example:



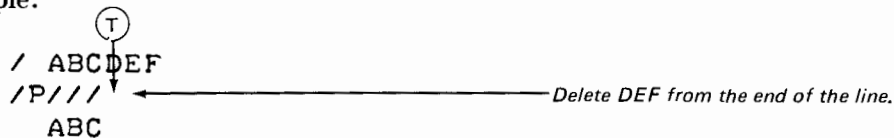
4-32. TRUNCATE CHARACTERS

The CTRL/T command is used to delete characters from the end of the pending line. When the control character is entered, the remainder of the line is eliminated.

Format:

```
PCTRL/T
CCTRL/T
OCTRL/T
```

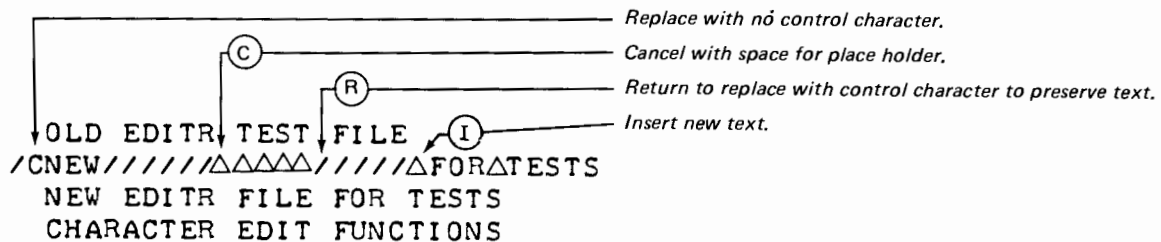
Example:



## EDITR Commands

### 4-33. CHARACTER EDIT COMBINATIONS

Several kinds of character editing can be done on one line of text. As an example; Replace, Cancel, Replace and Insert are shown. The control command advances the pending line after the edits.



### 4-34. CHARACTER EDIT SUMMARY

A summary of the ways in which characters can be replaced, inserted and deleted from a line of text is provided in Table 4-2.

Table 4-2. Character Edits Compared

MODE	REPLACE	CANCEL	INSERT
entered by	default R	C	I or S
prompt delimit- iter action	save current character	delete character	insert blank
tab action	spaces over saving characters	deletes to next stop	inserts to next stop
other characters	replace characters in same relative position	deletes characters	inserts characters

### 4-35. PATTERN EDITS

Pattern edits involve multiple lines of text from the source work area. There are two kinds of pattern edits: searches and exchanges.

A search matches a character string called a "find field" with a corresponding string in the source work area. There are four search commands (B, D, F, and J) which differ by where they begin the search and the length of the data block which they search.

Exchanges consist of two sets of character strings entered at the keyboard. Whenever the first string is encountered in the source work area, it is replaced by the second string. The commands used to perform exchanges are: G, Y, X, Z and U.

## 4-36. SEARCH COMMANDS

The B and J commands start searching at the beginning of the source work area. The D and F commands start searching at the pending line and search to a match or EOF. B implies that the source work area is replaced by the destination work area. The find field is only used as a key and is not actually manipulated as in the exchanges.

Find fields for search commands may consist of text, or of text preceded by the current delimiter, or an embedded escape character. The general form is:

```

B
F
D  [char[text]]
J  ───
    find field

```

The find field consists of a character string, or NULL. The editor will search for a line containing a matching field in the specified position(s). Only the number of characters specified are used in the match. If the find field is omitted, the last find field entered is used.

- ;
- The tab control character (; is the default), tabs the find field according to the established tab stops (columns 7 and 21 are the default stops). Tabbed over characters are replaced by blanks in the find field. For example:

```

      LABEL  LDA B,I
              LDA B,I

```

are instructions located in the source work area. The following search command will find all instructions containing blanks in columns 1-6 and LDA beginning in column 7.

```

/F;LDA

```

This command will find the second instruction, but will not find the instruction beginning with LABEL.

- esc
- The escape character indicates a field of indefinite length and may be used within a find field. For example:

```

/F esc XXX esc YYY

```

This command tells EDITR to find a line containing "XXX" followed by "YYY". "XXX" may be anywhere in the line and "YYY" is constrained only to follow the "XXX" but otherwise may be anywhere in the line. Note that on some terminals the escape key generates an action within the terminal that is not desirable. In this case, use the alternate escape key, ~ (sine wave).

- /
- Delimiter character (/ is the default), is used as the first character of the find field and indicates that the search for the match is limited to patterns beginning within the window established by the W command.

CTRL@ - Control @ as the find field will find a zero length line.



4-37. FIND A LINE WITH A FIND FIELD-SOF TO EOF

The B command searches the source work area, from the start-of-file to the end-of-file, for the first line of text which matches the find field. All lines passed over in the search are written to the destination work area; they are not deleted. When a line containing the correct character string is found, it becomes the new pending line. The search ends at EOF if no match is found. The B command always forces the destination work area to replace the source work area before the search.

Format:

B CTR/@	Search for zero length record
Btext	Search for left justified text
B(esc)text	Search for text anywhere in line
*B/text	Search for text in window
B;text	Search for field at tab stop
B	Successive search for same field. The same line will always be found by this command.

Example:

```

/BHRS ←————— Finds 1st left justified HRS.
HRS   BSS 1

(ESC)
/B(ESC)BSS ←————— Finds BSS anywhere in line.
HRS   BSS 1

/W7,9
/B/BSS ←————— Finds BSS in columns 7-9.
HRS   BSS 1

/B ←————— Find last specified field.
HRS   BSS 1
/B
HRS   BSS 1

/B;DEF ←————— Find first DEF at tab stop y.
      DEF HRS
    
```

\*/ in the command format B/text, is representative of the initial delimiter. If the delimiter has been changed (see 4-4), then the appropriate delimiter character must be used in place of the /. If the initial tab character (;) is changed (refer to Tab 4-6), the new tab character must be used.

4-38. FIND A LINE WITH A FIND FIELD FROM PENDING LINE TO EOF

This command causes a search of the source work area, beginning at the pending line and continuing to the end-of-file for a line containing a specified character string. When a line containing this field is found, it becomes the new pending line. If the line is not found, the search ends at the end-of-file.

Formats:

<i>Ftext</i>	Search for left justified field
<i>F(esc)text</i>	Search for field embedded anywhere in a line
<i>F/text</i>	Search for field within a window
<i>F;text</i>	Search for field beginning at tab stop
<i>F</i>	Successive searches for same field

Examples:

```

/1
  ASMB,R,L,T
/FDEC ←————— Left justified field.
EOF ←————— Not found.
  
```

```

/1
  ASMB,R,L,T
/FHRS ←————— Left justified field.
  HRS   BSS 1
  
```

```

(ESC) ASMB,R,L,T
/FDEC ←————— F(ESC)field.
  TCODE DEC 11
  
```

```

/1
  ASMB,R,L,T
/FERR ←————— Nonexistent field.
EOF
  
```

```

/5 ←————— Search from pending line.
  HRS   BSS 1
/FHRS
EOF
/4 ←————— Change pending line.
      EXT .ENTR,EXEC
/FHRS ←————— Left justify field.
  HRS   BSS 1
/5
  HRS   BSS 1
/FERR
EOF
  
```

## EDITR Commands

The F command followed directly by *text* searches for left justified occurrences of the field. F followed by the ESCAPE key and then *text* causes the field to be recognized anywhere in the line. If the requested field consists of CTRL/@, the EDITR searches for a zero length line.

Successive searches for the same character string do not require parameters. Once a pattern is entered for this command, it remains in effect until a new pattern is specified.

```
/W7,9
/F/BSS
  HRS   BSS  1
/F
  MIN   BSS  1
/F
  SEC   BSS  1
```

### 4-39. DELETE LINES TO FIND FIELD OR EOF

The D command deletes a block of text from the pending line to the line containing a specified field. After the deletion, the line of text containing the specified field becomes the new pending line. If the specified field is not encountered, the remainder of the file is deleted. If this command is used without specifying a field, the last field entered is used.

#### Format:

D CTRL/@	Zero length field
D <i>text</i>	Left justified field
D $\textcircled{\text{esc}}$ <i>text</i>	Delete to field anywhere in line
D/ <i>text</i>	Delete to field in window
D	Successive deletes
D; <i>text</i>	Delete to field beginning at tab stop

#### Example:

To delete to a left justified field:

```
0001  ASMB,R,L,T
0002      NAM TIME
0003      ENT WEN
0004      EXT ,ENTR,EXEC
      :
0020      STA SEC,I
0021      JMP WEN,I
0022  TCODE DEC 11
0023  TBUFF BSS 5
0024      END
```

} Original source work area text.

```

/1
ASMB,R,L,T
/DTCODE ← Deletes from first line to first left justified occurrence of TCODE field.
TCODE DEC 11 ← New pending line displayed.
/1
TCODE DEC 11 ← Verify that new pending line is also the first line of text.
/L7 ← List contents of edited source work area.
TCODE DEC 11
TBUFF BSS 5
      END
EOF

```

To delete to first occurrence of MIN field anywhere in a line (D  $\text{\textcircled{esc}}$  text):

```

0001 ASMB,R,L,T
0002     NAM TIME
0003     ENT WEN
0004     EXT ,ENTR,EXEC
0005 HRS  BSS 1
0006 MIN ← BSS 1
0007 SEC  BSS 1

0017     LDA TBUFF+2
0018     STA MIN,I
0019     LDA TBUFF+1
0020     STA SEC,I
0021     JMP WEN,I
0022 TCODE DEC 11
0023 TBUFF BSS 5
0024     END

```

Original text in source work area contains two instances of find field.

```

 $\text{\textcircled{ESC}}$  ASMB,R,L,T
/DMIN ← Pending line is the first line in the source work area.
MIN  BSS 1
/1 ← Delete to first occurrence of MIN anywhere in a line.
MIN  BSS 1 ← New pending line containing find field displayed.
/L9 ← List contents of edited source work area after pending line to show
      nothing else deleted.
MIN  BSS 1
SEC  BSS 1
WEN  NOP
      JSB .ENTR
      :

```

## EDITR Commands

To delete to a field within a window (D/text):

```

0001  ASMB,R,L,T
0002      NAM TIME
0003      ENT WEN
0004      EXT ,ENTR,EXEC
0005  HRS  BSS 1
0006  MIN  BSS 1
0007  SEC  BSS 1

0017      LDA TBUFF+2
0018      STA MIN,I
0019      LDA TBUFF+1
0020      STA SEC,I
0021      JMP WEN,I
0022  TCODE DEC 11
0023  TBUFF BSS 5
0024      END

```

Two occurrences of find field.  
Window will be set to exclude one occurrence.

```

/W11,13
/D/MIN
    STA MIN,I
/1
    STA MIN,I
/L20
    STA MIN,I
    LDA TBUFF+1
    STA SEC,I
    JMP WEN,I
TCODE DEC 11
TBUFF BSS 5
    END
EOF

```

Window set to include only second occurrence of MIN.  
New pending line displayed.  
Verify first line and pending line are the same.  
List contents of edited source work area to verify deletion of text to second occurrence of MIN.

Delete to field beginning at first tab stop:

```

/D;STA
    STA HRS,I

```

Delete to successive occurrences of the find field (D):

```

SOURCE FILE?
/TIME
(ESC) ASMB,R,L,T
/DMIN
    MIN  BSS 1
/D
    STA MIN,I
/D
EOF

```

First delete with (esc) field.  
Successive deletes use the same window and same field until reset. Delete to MIN in any position. Display new pending line.  
No more occurrences of MIN before end-of-file.

## 4-40. JUMP TO FIND FIELD LINE AND MAKE IT NEW PENDING LINE

The J command jumps, without putting the jumped over lines into the destination work area, to the first line of the source work area containing a specified character string and makes it the new pending line. The old pending line is always saved (written to the destination work area), before the jump is made. The jump is always made to the first line containing the match. This command can be used to either delete or copy lines from the source work area.

Format:

<i>Jtext</i>	Jump to left justified field
<i>J<sup>(esc)</sup>text</i>	Jump to field embedded in line
<i>J/text</i>	Jump to field in window
<i>J</i>	Successive jumps to same field (not useful since the jump is always made to the first line)

Example:

```

/JHRS ←————— Jump to left justified field.
HRS   BSS 1

/W7,9
/J/BSS ←————— Jump to line with field in window.
HRS   BSS 1

(ESC) ————
/JBSS ←————— Jump to field anywhere in line.
HRS   BSS 1

/J
HRS   BSS 1 ←————— Successive jumps to the same field.
/J
HRS   BSS 1

```

Successive jumps through the source work area to the same pattern cannot be made. Since the destination area always replaces the source work area, when this command is entered, only the first instance of the input string will ever be matched, as shown in the last example.

You can use the J command to save the pending line and delete to a pattern. The deletions begin with the line following the pending line. The deletions cover a block of text up to the line containing the specified pattern.

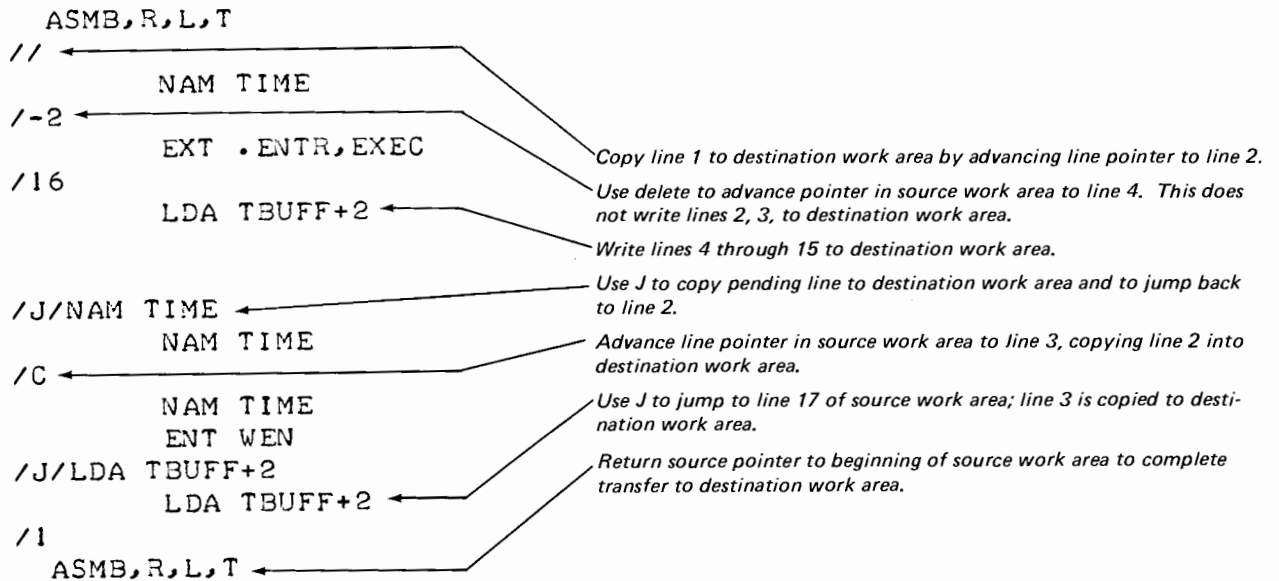
To use the J command to move lines of code from one place to another as in the example which follows: 1) the code is deleted from its original location in the destination work area, and 2) the lines of code are copied from the source work area into the new location in the destination work area.

File Manager listings output on the line printer are used in this example because the sequence numbers make the example easier to follow. The J command does not add numbers to the code.

```

0001  ASMB,R,L,T
0002      NAM TIME ]
0003      ENT WEN ]
0004      EXT .ENTR,EXEC
0005  HRS   BSS 1
0006  MIN   BSS 1
0007  SEC   BSS 1
0008  WEN   NOP
0009      JSB .ENTR
0010      DEF HRS
0011      JSB EXEC
0012      DEF *+3
0013      DEF TCODE
0014      DEF TBUFF
0015      LDA TBUFF+3
0016      STA HRS,I
0017      LDA TBUFF+2
0018      STA MIN,I
0019      LDA TBUFF+1
0020      STA SEC,I
0021      JMP WEN,1
0022  TCODE DEC 11
0023  TBUFF BSS 5
0024      END
    
```

Move lines 2 and 3 so that they follow line 14.



```

0001  ASMB,R,L,T
0002  EXT  .ENTR,EXEC
0003  HRS   BSS 1
0004  MIN   BSS 1
0005  SEC   BSS 1
0006  *EN   NOP
0007  JSB  .ENTR
0008  DEF  HRS
0009  JSB  EXEC
0010  DEF  *+3
0011  DEF  TCODE
0012  DEF  TBUFF
0013  LDA  TBUFF+3
0014  STA  HRS,I
0015  NAM  TIME
0016  ENT  *EN
0017  LDA  TBUFF+2
0018  STA  MIN,I
0019  LDA  TBUFF+1
0020  STA  SEC,I
0021  JMP  *EN,I
0022  TCODE DEC 11
0023  TBUFF BSS 5
0024  END

```

Lines of code deleted from here.

Lines of code inserted here.



#### 4-41. EXCHANGE COMMANDS

All of the exchange commands use find fields to locate where the exchange will be made. The character string in the find fields of the G, Y, X, and Z commands function as literals; each character is matched with a character in the source work area. The characters in the find fields of the U and V commands are used as place holders. The find field is used to specify the number of characters to be replaced. The general form is:

G }  
Y } *old string/new string*  
X }  
Z }

U }  
V } *character position/new string*

Remember that the find fields are separated by the delimiter character. If this character is changed with the X command, the new one must be used here.



#### 4-42. CHARACTER REPLACE ON PENDING LINE

The G command performs an immediate exchange of an existing character string with a new character string. It performs the exchange on the pending line, displays the result, and the pending line remains the pending line.

The old string may be any length except zero, and the new string any length including zero. Exchanges are made only if the first character of the old string is within the window established by the W command. An exchange pattern remains in effect until a new one is entered.

Format:

*Gold string/new string*

Example:

```
/W7,9
/5
  HRS   BSS 1
/GBSS/CRG
  HRS   CRG 1
```

The exchange fields used in the G command are not tabbed, so that the tab character can be used like any other character in the exchange string.

The only time the current delimiter is recognized as a delimiter is the first time it occurs separating the old string and the new string. Any subsequent delimiters are treated as part of the "new" character string. If the delimiter between strings is omitted, EDITR returns the error message ??.

## 4-43. EXCHANGE ON PENDING LINE, DISPLAY NEXT OCCURRENCE OF PATTERN

The Y command performs an immediate exchange between an existing character string and a new character string. The exchange is always made on the pending line, but after the exchange, the next occurrence of the old string automatically becomes the new pending line, and eligible for an exchange. The old string may be any length, except zero, and the new string may be any length, including zero. Exchanges are made only if the first character of the first string is within the window established by the W command.

Format:

*Yold string/new string*

Example:

```

/W7,9
/5
  HRS   BSS 1
/YBSS/CRG
  HRS   CRG 1
  MIN   BSS 1

```

*Result of exchange displayed.*

*Next occurrence of BSS in columns 7-9.*

After the first exchange, successive exchanges are made by entering Y with no parameters. The exchange pattern, once entered, remains in effect until a new one replaces it.

```

/Y
  MIN   CRG 1
  SEC   BSS 1
/Y
  SEC   CRG 1
  TBUFF BSS 5
/Y
  TBUFF CRG 5
EOF

```

*Result of exchange.*

*Next occurrence of old string.*

A combination of Y and F commands can be used to do selective exchanges. The F command is entered (without parameters) to skip an exchange on this occurrence and find the next occurrence.

```

/5
  HRS   BSS 1
/YBSS/CRG
  HRS   CRG 1
  MIN   BSS 1
/F
  SEC   BSS 1
/Y
  SEC   CRG 1
  TBUFF BSS 5
/F
EOF

```

*Replace BSS with CRG in columns 7-9.*

*Display next occurrence of pattern.*

*Use F to find next occurrence of old string in window columns 7-9.*

The exchange fields used in the Y command are not tabbed, so that the tab character can be used like any other character in the exchange string.

Only the first occurrence of the current delimiter is recognized. It is assumed to separate the old string and the new string. Any subsequent delimiters are treated as part of the "new" character string.

4-44. ENABLE EXCHANGE PATTERN ALL LINES (LIST)

This command can be used to make exchanges over a variety of ranges. It is always used in combination with a second command which defines the range. For example, F or B. The X command performs an exchange of an old character string for a new character string. The old string can be any length except zero. The new string can be any length including zero. The exchange takes place when the next command is executed. All lines where an exchange takes place are printed. The print device is usually the console, but may be some other device if the second command is D, n, / or + which specifies an *lu* parameter. This *lu* is then used as the print device. The exchange takes place only if the first character of the old string is within the window established by the W command.

The number of lines in which exchanges occur depends on which range command is specified, but generally includes all lines passed over by a search or by positioning to a new pending line. The exception is when the B command is used, and the destination work area replaces the source work area. The exchanges take place between the pending line and EOF. The exchange is then disabled for the search from the beginning of the source work area.

The exchange takes place from the pending line to the EOF. In all cases the exchange is cancelled after completion of the second command. To do a second exchange with the same patterns, the exchange must be re-enabled using X.

Format:

*Xold string/new string*  
*range command*

Example:

```

/W 7,9
/XBSS/CRG
/BBSS
HRS   CRG 1
MIN   CRG 1
SEC   CRG 1
TBUFF CRG 5
EOF
    
```

← Old string = BSS, new string = CRG.  
 ← B command defines range from here to end-of-file.  
 ← All lines where exchanges took place.

4-45. ENABLE EXCHANGE PATTERN ALL LINE (NO LIST)

The Z command is used for the same purpose as the X command. The only difference between these two commands is that X displays all lines in which exchanges take place; the Z command does not. The obvious advantage of using the Z command is that you do not have to wait for a system response before continuing to the next exchange.

Format:

*Zold string/new string*  
*range command*

For example, see X command (4-42).

## 4-46. UNCONDITIONAL CHARACTER REPLACE (LIST)

This command is used to insert, delete, or replace the number of characters in the old string with a new string. The characters in the old string are replaced by the new string. The exchange field starts at the first position of the window, but is not limited by its length. The format of this command determines its function as follows:

- inserts    — first field omitted
- deletes    — second field omitted
- replace    — both fields greater than zero

**Format:**

*Old string/new string*

Where:

*old string* is a maximum of 148 alphanumeric characters.

The characters in the old string are replaced beginning at the first column of the window. The old string is not used as a pattern for a search, but rather, to specify the number of characters to be replaced at the start of the window by the new character string. Lines in which the exchange is made are displayed.

**Example:**

Change the first character of every line in a file to a File Manager command to list (:L1):

	/W1	←	Set window to begin at first column
ESC	/V1/:L1	←	Replace first character with :L1,
	/FEND	←	Do every line until END found

## EDITR Commands

### 4-47. UNCONDITIONAL CHARACTER REPLACE (NO LIST)

The U command functions exactly like the V command, but does not display the results of the exchanges.

Format:

*Uold string/new string*

Example:

This example takes a file formatted for disc storage and reformats it for storage on magnetic tape. Output is from the File Manager :LI command since neither the Z or U commands list.

#### PARTIAL LIST OF ORIGINAL SOURCE

```
/ 0169 :ST,8,#LOADR : 17738::7:-1,BA
/ 0170 :ST,8,#TASMB : 17738::7:-1,BA
/ 0171 :ST,8,#FTNC1 : 17738::7:-1,BA
/ 0172 :ST,8,#FTNC2 : 17738::7:-1,BA
/ 0173 :ST,8,#RTSGN : 17738::7:-1,BA
/ 0174 :ST,8,#ELQUI : 17738::7:-1,BA
/ 0175 :ST,8,#RTBTG : 17738::7:-1,BA
/ 0176 :ST,8,#LABSYS: 17738::7:-1,BA
/ 0177 :ST,8,LBPRI : 17738::7:-1,BA
/ 0178 :ST,8,LBMEM : 17738::7:-1,BA
```

## EDIT COMMANDS

```
/Z:ST,8/:DU, ← Exchange every occurrence of old string ":ST,8," for new string ":DU,,".
/9999 ← The range command could be any large number sufficient to insure the
EOF exchange takes place in every record of the file.

/W19,150 ← Set window to begin at the start of the replace field.
/U1234567890/,8 ← Replace whatever exists, beginning at column 19 and extending 10 col-
umns, with the field "8".
/9999 ← Range command is, again, any number large enough to insure replace
EOF takes place on every record in the file.
```

## DESTINATION AREA

### EDITR RESULTS

```
0069 :DU,#LOADR : 17738,8
0070 :DU,#TASMB : 17738,8
0071 :DU,#FTNC1 : 17738,8
0072 :DU,#FTNC2 : 17738,8
0073 :DU,#RTSGN : 17738,8
0074 :DU,#ELQUI : 17738,8
0075 :DU,#RTBTG : 17738,8
0076 :DU,#LABSYS: 17738,8
0077 :DU,LBPRI : 17738,8
0078 :DU,LBMEM : 17738,8
```

#### 4-48. TERMINATE COMMANDS

The terminate commands are used to end EDITR operations. With the exception of the ABORT command, they perform any enabled exchanges, cause data remaining in the source work area to go to the destination work area, and cause the edited destination work area to be written to the output file.

All of the normal EDITR termination commands begin with E. Once E is entered EDITR begins its termination process. Only the second letter of a normal terminate command or rubout may be entered at this time.

Output to a device of a type 0 file can only be done using the ER or ELR terminators. This assumes that a directory entry has already been created with a file name and a file type designation of 0, and the appropriate legal subfunctions for the device being used.

#### 4-49. ABORT

This command is used to end EDITR operations and leave the original source file unchanged.

Format:

A

Example:

In this example, EDITR is deliberately aborted by the user:

```

/1
  ASMB,R,L,T
/5
  HRS    BSS 1
/L3
  HRS    BSS 1
  MIN    BSS 1
  SEC    BSS 1
  WEN    NOP
/A
EDITR ABORTED

```

## EDITR Commands

### 4-50. ASSIGN EDITED FILE TO LS TRACKS (EL)

This command ends the editing session and assigns the contents of the edited destination area to the RTE logical source (LS) tracks. The location of the LS tracks is returned to the user, and the logical source pointer is set by EL so that the system LS command is not needed.

Format:

```
EL
```

Example:

```
:RU, EDITR
SOURCE FILE?
/Δ ← EDITR creates the file in the Logical Source (LS) tracks.
EOF
/ ASMB, R, L, T
/      NAM TIME
/      ENT WEN
/      EXT .ENTR, EXEC
/EL
  LS FILE 2 30 ← The edit is terminated and the file stored in the LS tracks with the EL
END OF EDIT      command.
```

```
:RU, EDITR ← Run EDITR again
SOURCE FILE?
/Δ ← EDITR copies LS tracks into its source work area
  ASMB, R, L, T
/L10 ← List 10 lines
  ASMB, R, L, T
      NAM TIME
      ENT WEN
      EXT .ENTR, EXEC
EOF
      These are lines put into LS tracks by EL command above.
```

This command should be used with some caution if the file came from the LS area, because once the destination file is moved to the logical source area it replaces the original text in that area and there is no longer a back up copy of the source file. This source file can be saved with the ELC or ELR EDITR commands (see 4-51, 4-52).

## 4-51. END EDIT AND CREATE A FILE MANAGER FILE (EC)

This command ends the edit and creates a named File Manager file for storage of the destination work area.

Format:

```
ECname[:security code[:cartridge reference number]]
```

where:

*name* = 1-6 character file name  
*security code* = positive or negative decimal integer or two ASCII characters; it may be omitted.  
*cartridge reference number* = negative logical unit number of cartridge or positive integer identifying cartridge; it may be omitted.

Example:

In this example, a file is created on-line by EDITR in the Logical Source area:

```
:RU,EDITR
SOURCE FILE?
/Δ
EOF
/IASMB,R,L,T
/      NAM TIME
/      ENT WEN
/      EXT .ENTR,EXEC
/ HRS  BSS 1
/M

/ MIN  BSS 1
/ SEC  BSS 1
/ WEN  NOP
/      JSB .ENTR
/      END
/ECCAROL:LS:-2
END OF EDIT
```

The EC terminate command is used to Create a File Manager file named CAROL. The security code for the file is LS. The negative value in the Cartridge Reference Number parameter points to a logical unit number (in this case, the system disc). The Batch-Spool Monitor Manual contains a full description of the use of negative cartridge reference values.

A File Manager listing of the file just created with the EC command follows:

```
CAROL T=00004 IS ON CR00002 USING 00002 BLKS R=0010

0001 ASMB,R,L,T
0002      NAM TIME
0003      ENT WEN
0004      EXT .ENTR,EXEC
0005 HRS  BSS 1
0006 MIN  BSS 1
0007 SEC  BSS 1
0008 WEN  NOP
0009      JSB .ENTR
0010      END
```



## EDITR Commands

### 4-52. REPLACE OLD FILE WITH NEW FILE (ER)

This command ends the edit and replaces an old File Manager file with a new file. You can replace the source file or another file you designate with this command.

#### Format:

<p>ER = This form can only be used if the source file came from a File Manager file originally.</p> <p>ERname = It cannot be used if the source file originally came from the LS area.</p> <p>name = 1-6 character file name.</p>
---

#### Example:

Use ER to end edit and replace the original source file with edited text:

```
:CR, TIME:::3:12
:RU, EDITR
SOURCE FILE?
/TIME
EOF
/ ASMB, R, L, T
/     NAM TIME
/     ENT WEN
/     EXT .ENTR, EXEC
/ HRS  BSS 1
/ER
```

← Create source file named TIME, using the File Manager. This command only creates a file name, no data is transferred.

← The text from the EDITR destination work area is transferred into the File Manager file.

Use ERname to end edit and replace the data in file name with the edited text.

```
:CR, CRIME:::3:12
:RU, EDITR
SOURCE FILE?
/TIME
  ASMB, R, L, T
/LIST
  ASMB, R, L, T
    NAM TIME
    ENT WEN
    EXT .ENTR, EXEC
  HRS  BSS 1
EOF
/ERCNAME
```

← Create CRIME, file name with the File Manager.

← List the contents of TIME.

← End edit; Replace contents of CRIME with edited file.

END OF EDIT

## 4-53. END EDIT, CREATE FILE MANAGER FILE, STORE IN LS TRACKS (ELC)

This command ends the edit, creates a File Manager file and stores the edited text both in the newly created file and in the system LS tracks.

Format:

```
ELCname  [:security code[:cartridge reference]]
```

(See paragraph 4-51 for syntax definitions)

Example:

```
SOURCE FILE?
/
ASMB,R,L,T
/L9
ASMB,R,L,T
      NAM TIME
      ENT WEN
      EXT .ENTR,EXEC
HRS   BSS 1
MIN   BSS 1
SEC   BSS 1
WEN   NOP
      .END
EOF
/ELCCAROL2:LS:2
  LS FILE 2 36
END OF EDIT
```

*The security code and cartridge reference are supplied in this example. If they are omitted, they would default to 0.*

*The logical source location is returned so that the pointer can be set for an assembly or compilation.*

## EDITR Commands

### 4-54. END EDIT, REPLACE FILE AND NAME, AND STORE IN LS TRACKS (ELR)

This command ends the edit, replaces an existing File Manager file under the original name or a new name if supplied, and stores the same data in the LS tracks.

#### Format:

```
ELR
or
ELRname[:security code[:cartridge reference number]]
(See paragraph 4-51 for syntax definitions)
```

```
/L10
  FTN,L
  C
  C    MAIN PROGRAM
  C
      PROGRAM PRIME
      DIMENSION I(5)
      WRITE(6,100)
100  FORMAT(12X,"PRIMES FROM ONE TO FIVE HUNDRED TEN"////
          19X,"1",19X,"2",19X"3")
      J=4
      M=1
/ELRPRIME
  LS FILE 2 30
END OF EDIT
```

```
/7
  SEC    BSS 1
/L5
  SEC    BSS 1
  WEN    NOP
      JSB  .ENTR
      DEF  HRS
      JSB  EXEC
      DEF  *+3
/ELR ←
```

*In this example the old File Manager file is replaced by a new file with the same name. A file is not created with this command. The old file contents are written over with the contents of the edited destination work area.*

*If a security code was used with the original file name, this form of the EL terminator will cause an FMGR-7 error when EDITR attempts to write new contents into the old file. The ELR name form must be used with the security code.*

\*LG, 1

# EDITR IN BATCH ENVIRONMENT

SECTION

V

*had recommended this*

In batch mode, commands to EDITR are supplied as part of the job command file; that is, all EDITR commands must be supplied before the job is begun. Once the job is under way, you can not interact with the computer to change the course of the processing.

All EDITR commands function the same in batch as they do in the interactive mode. You need not supply the EDITR prompt (/) on the input commands.

It is important that you know beforehand what is going to result from the edit commands in the job deck. Any condition which causes an error message causes EDITR to abort since there is no possibility of user intervention to fix the error. You must also insure that EDITR goes through the normal terminate sequence initiated by EL, EC, ER, ELR, or ELC. If an edit deflects this sequence, EDITR aborts.

If EDITR is run in batch mode without spooling, File Manager commands and EDITR commands must be read from an external device. Under spooling, however, EDITR commands can be read from a file if the file appears to be an external device. This is done by setting up an input spool to reference a file name as shown in the examples which follow. The default attributes should always be specified for the input spool.

## 1) EDITR WITHOUT SPOOLING

Card reader defined as LU #5

```
CONSOLE: *RU,FMGR,5
          :RU,EDITR,5
          { Job deck in card reader }
```

## 2) EDITR WITH SPOOLED JOB

```
:JO,CAROL
```

```
:RU,EDITR,5
```

```
editor commands
```

```
.  
. .  
. .
```

```
ECname
```

```
:EOJ
```

*The EDITR commands appear in the job stream which may be a file or a card deck, or tape, etc.*

```
:JO,CAROL2
```

```
:LU,50,COMND
```

```
:RU,EDITR,50
```

```
:EOJ
```

*Set up input spool to reference file COMND containing the EDITR commands.*

*In this example, EDITR acts as if it were reading from LU50, but it is actually reading records from COMND through the spool monitor driver.*



# EDITR WITH MULTI TERMINAL MONITOR


SECTION

VI

The Multi-Terminal Monitor (MTM) allows any terminal to act as though it exclusively owns the programs it executes. Several users then can run the same program at the same time if each terminal has a separate copy of the program. The RTE manuals contain a description of the use of EDITR in this environment so that only the directions for making program copies are included here. Several special considerations apply to MTM under RTE-IV as explained below.

To create several copies of the EDITR, use the SP command to create a copy of EDITR as a FMGR file. Then use the RN (rename) command to rename the file, and the RP (restore program) command to build an ID segment referencing the file by the new name. A sequence of commands to do this is shown below.

*RU, FMGR	←	Run File Manager.
: SP, EDITR	←	Save a copy of EDITR in a file.
: RN, EDITR, ED07	←	Rename the file to ED07
: RP, ED07	←	Create an ID segment pointing to the file.
: RN, ED07, ED08	←	Rename the file to ED08
: RP, ED08	←	Create an ID segment pointing to this file.
:		Continue this pattern until you have created as many names as you want.
:		
: RN, ED22, EDITR	←	Change the final name back to EDITR for the next run of this procedure.
: TR	←	Then terminate FMGR.



Any of the renamed programs may now be run using the RU command.

The code to run the File Manager and save a copy of EDITR in a file is used only once in the life of the system. The rest of the code can be stored in a transfer file which is then run each time the system is rebooted.

Since every user has equal access to the LS area care must be taken storing anything in this area.

## 6-1. RUNNING EDITR UNDER RTE-IV MULTI-TERMINAL MONITOR

In an RTE-IV Multi-Terminal Monitor (MTM) environment, MTM processes ID segments to perform many of the above actions automatically. Under the circumstances described below, a copy of the program that "belongs" to the terminal is automatically made for the user when he gives the RUN command. This allows several users to simultaneously run the same program, each from his own copy. Refer to the MTM section of the RTE-IV Programmer's Reference Manual for more information.

Under the following conditions, MTM will make a copy of the EDITR named EDLxx and schedule it to the terminal with logical unit number xx:

- 1) it's an RTE-IV Operating System
- 2) the user is operating at an MTM terminal of logical unit number xx.
- 3) FMGxx is the father (the user is operating under program FMGxx).
- 4) EDITR is a temporary program (i.e. it was loaded on-line with the LOADR. If it was loaded at generation, it can be purged and reloaded with the LOADR).

## EDITR with Multi Terminal Monitor

If the above conditions are met, the user need only schedule the EDITR using the RUN command as follows:

```
:RU,EDITR
```

A copy of the EDITR named EDLxx will be created and scheduled to terminal xx.

# EDITR IN MULTIPOINT ENVIRONMENT

SECTION

VII

## 7-1. INTRODUCTION

In a multipoint environment, several special considerations apply to EDITR operations. A terminal is defined to be in a multipoint environment if its I/O is being processed through driver DVR07. To determine if a terminal is operating under multipoint, observe the transmit break light on the terminal. If it is blinking intermittently, the terminal is probably a multipoint terminal.

When the EDITR is invoked from a multipoint terminal, several actions are performed for the user. The intrinsic tab function of the terminal is enabled (the EDITR's tab character (the semicolon) remains on). The INSERT CHAR, DELETE CHAR, and CLEAR DISPLAY keys on the terminal are enabled for use as editing functions (Note that the EDITR commands to insert and delete lines should be used, not the INSERT LINE and DELETE LINE keys on the terminal). Finally, since the CONTROL and ESCAPE keys do not work in a multipoint environment, the Q and O commands rather than the edit commands explained in the preceding sections of this manual should be used to perform character edits. All of the other EDITR commands may be used in a multipoint environment.

It should be remembered that in a multipoint environment the ENTER key, and not the carriage return, is the key used to transmit text to the multipoint controller. Furthermore, the characters that the EDITR will process are usually delimited by the left margin on the left and the current cursor position on the right. Paying attention to these observations will help the user greatly.

## 7-2. Q AND O COMMANDS

The Q and O commands are used for character edits in a multipoint environment. Both commands are used to edit the pending line. The only difference between them is that the O command immediately sends a copy of the pending line to the destination work area while the Q command does not. Although only the Q command is discussed throughout the rest of this section, the discussion applies to the O command as well.

When the Q command is entered, the pending line is displayed, along with a delimiter (GS) to the left of the line. The delimiter is not part of the text string and must be preserved to assure proper operation. The delimiter is represented as a pound sign (#) throughout the rest of this section. The EDITR will position the cursor underneath the first character of the line. You may now edit the line using any of the following procedures.

To retain the pending line as it is, immediately hit the ENTER key. For example:

```
/Q
#ABCDEFGHIJKL ← Cursor displayed under first character in line.
-
/P
ABCDEFGHIJKL ← Line displayed remains the same
/
```

*Press the ENTER key and line is retained as is.*



To truncate characters from the end of a line, position the cursor immediately after the last character to be retained. Strike the ENTER key to enter all the characters between the left margin and the current cursor position. The intrinsic terminal key CLEAR DISPLAY can also be used to delete characters at the end of a line. After using the CLEAR DISPLAY key, the ENTER key is used to enter the edited line. For example:

```

/Q
#ABCDEFGHIJKL ← Position cursor under I, press ENTER key
  ABCDEFGH ← Edited line is displayed
/Q
#ABCDEF_GH ← Position cursor under F, press CLEAR DISPLAY, then ENTER
  ABCDE ← Edited line is displayed
/
    
```

To add characters in the middle of a line, the INSERT CHAR key may be used. Press the INSERT CHAR key. The red light above the key should come on. Move the cursor to the position where the characters are to be added, and type in the new characters. Finally, position the cursor at the end of the line and hit the ENTER key. The insert light will go off and the edited line will remain as the pending line. For example:

```

/Q
#ABCDEFGHIJKL ← Position cursor underneath F, depress the INSERT CHAR
  - key, and type in the new characters "123". Position
  ABCDEF123GHIJKL ← Edited line is displayed
/
    
```

To delete one or more characters, position the cursor under each character to be deleted and press the DELETE CHAR key. The character will be deleted from the display and the rest of the line will be shifted left to fill in the gap. After all of the desired deletions have been made, move the cursor to the end of the line and press the ENTER key. Do not delete the delimiter at the beginning of the line. For example:

```

/Q
#ABCDEFGHIJKL ← Position cursor under the G, press the
  - DELETE CHAR key three times, move the cursor
  ABCDEFJKL ← Edited line is displayed
/
    
```

### 7-3. TAB CONTROL IN MULTIPOINT ENVIRONMENT

When the EDITR is invoked in a multipoint environment, the TAB key on the terminal is enabled (the EDITR tab character remains on). The tab stops are initially set to columns 7 and 21. These may be changed using the SET TAB and CLEAR TAB keys on the terminal. The TAB key may be used to position the cursor at any time. It is equivalent to moving the cursor using the terminal keys with arrows on them.

# SAMPLE EDIT

APPENDIX

A

This material is provided to show some of the EDITR commands used in combinations. The material is divided into two parts: 1) creation and edit of a comment file called COMENT and a FORTRAN program, &FLIST, and 2) the compiler output showing the complete program results of the edits and the merging of the two files.



## EDIT SESSION

```

Ø7>ON,FMG7
:CL
:CR,COMENT:::3:12 ← Create file COMENT.
:CR,&FLIST:::3:12 ← and file &FLIST.
:LS,Ø
:RU,EDIØ7
SOURCE FILE?
/Δ ← Create source area on-line for COMENT file.
EOF
/T;4 ← Set first tab to column 4 to position comments.
/ ;
/ ;
/ ;
/Ⓞ ← Delete bell ring.
/ ;THE PROGRAM FLIST READS A TYPE 3 OR 4 FILE MANAGER FILE
/ ;AND PERFORMS A LIST FUNCTION (SAME AS ':LI,NAMR') TO THE
/ ;LINE PRINTER WITH LINE NUMBERS IN THE FIRST FIVE COLUMNS.
/ ;
/ ;THE SEQUENCE OF OPERATIONS FOLLOWS:
/GFIL/FOL ← Replace characters in line just created.
    THE SEQUENCE OF OPERATIONS FOLLOWS: ← Display result of edit.
/ ;
/ ;(1)  GET PARAMETERS AND SET UP DEVICES,DEFAULTING
/ ;    IF NECESSARY.
/ ;(2)  GET FILE NAME FROM USER AND OPEN IT.
/ ;(3)  IF FILE ERROR,INFORM USER AND QUIT.
/ ;(4)  IF FILE TYPE NOT 3 OR 4, INFORM USER AND QUIT.
/ ;(5)  INITIALIZE LINE COUNT.
/ ;(5)  READ A LINE FROM THE FILE.
/G5/6 ← Change number with character replace on pending line.
    (6)  READ A LINE FROM THE FILE. ← and display corrected line.
/ ;(7)  IF END OF FILE,WRITE MESSAGE AND QUIT.
/ ;(8)  IF READ ERROR,WRITE MESSAGE AND QUIT.
/ ;(9)  LIST THE LINE ON THE LIST DEVICE.
/ ;(10) UPDATE THE LINE COUNT AND GO BACK TO STEP (5).
/ERCOMENT ← Store contents of destination work area into COMENT file created
END OF EDIT ← at the beginning of the session.
:LI,COMENT ← COMENT listed on line printer – see lines 2-21 of &FLIST listing.
:RU,EDIØ7
SOURCE FILE?
/Δ ← Request LS area.
EOF
/ FTN4,L,B
/MCOMENT ← Merge COMENT file into new file being created on-line.
EOF
/Ø ← Display line 1 of source work area.
    FTN4,L,B

```

## EDIT SESSION (Continued)

```

/C ←————— Advance pending line to next line before "C's" are inserted.
  FTN4,L,B

/W1,1 ←————— Set window for first column.
/X /C ←————— Exchange every blank column 1 for a "C".
/FXX ←————— Find a nonexistent pattern. The purpose of this command is to be sure
  C                                     the exchange is made on every record in the file.
  C
  C
  C THE PROGRAM FLIST READS A TYPE 3 OR 4 FILE MANAGER FILE
  C AND PERFORMS A LIST FUNCTION (SAME AS ':LI,NAMR') TO THE
  C LINE PRINTER WITH LINE NUMBERS IN THE FIRST FIVE COLUMNS.
  ●
  ●
  ●
EOF
/C ←————— Insert text after the pending line.
/C THE PROGRAM STARTS HERE.
/C
/ ;PROGRAM FLIST(3,90)
/C
/ ;INTEGER DCB(144,LUN(5),INBUF(40),FNAME(3)
/C
/ ;CALL RMPAR(LUN)
/ ;LUTTY=1
/ ;LULST=6
/ ;IF(LUN(1).NE.0) LUTTY=LUN(1)
/ ;IF LUN(2).NE.0) LULST=LUN(2)
/W1,150 ←————— Reset windows to maximum value.
/GIF/IF( ←————— G command used to insert "{".
  IF LUN(2).NE.0) LULST=LUN(2)

/ 10 FNAME(2)=2H
/P ←————— Request pending line display.
  10 FNAME(2)=2H
/G(2/(1 ←————— G used to reset FNAME value from 1 to 2.
  10 FNAME(1)=2H
/O ←————— O used to duplicate FNAME line and change value to 2.
  /////2
  FNAME(2)=2H
/O ←————— Duplicate FNAME(2) line.
  FNAME(2)=2H
/G(2/(3 ←————— Replace characters "2" with "3".
  FNAME(3)=2H
/ ;WRITE(LUTTY,100)
/ ;READ(LUTTY,101)FNAME
/ ;CALL P-OPEN(DCB,IERR,FNAME,0) ←————— CTRL/A used to delete space (echoed as a back arrow).
/ ;IF(IERR.EQ.3.OR.IERR.EQ.4) GO TO 20
/I;IF(IERR.GE.0) GO TO 50 ←————— Insert text before pending line.
/I;IFWRITE(LUTTY,102)FNAME,IERR
/ ;WRITE(LUTTY,106)FNAME
/ ;GO TO 999
/ 20 LINE=1
/ 30 CALL READF(DCB,IERR,INBUF,40,LEN)
/ ;IF(LEN.EQ.-1) GO TO 70
/ ;IF(IERR.EQ.0) GO TO 40
/ ;WRITE(LUTTY,103)FNAME,IERR

```

EDIT SESSION (Continued)

```

/ ;GO TO 999
/ ;WRITE(LULST,104)LINE,(INBUF(K),K=1,LEN)
/ ;LINE=LINE+1
/ ;GO TO 30
/ 70 WRITE(LULST,105)

/ΔC←----- Insert blank comment line.
/Δ;F\
  100 FORMAT(// " ENTER FILE NAME: ")
/ 100 FORMAT(3AS2)
/G100/101←----- Character replace to change statement number.
  101 FORMAT(3AS2)
/ 102 FORMAT("FILE ""3A2"" ERROR: ""17/")
/ 103 FORMAT("FILE ""3A2"" READ ERROR: ""17/")← Use RUBOUT key here.
/ 104 FORMAT(X,14,X,37←←40A2
/ 105 FORMAT(// " END-OF-FILE"/"1")
/ 106 FORMAT("FILE ""3A2"" OF WRONG TYPE")
/ΔC←----- Insert comment line.
/ 999 END
/ ;END$
/C←----- Advance pending line.
      END$
EOF
/ER&FLIST←----- Store combined source in File Manager created file.
END OF EDIT

```

```

:RU,EDI07
SOURCE FILE?←----- Second editing pass used to clear up known errors.
/&FLIST
  FTN4,L,B
/41
      IFWRITE(LUTTY,102)FNAME,IERR
/ ;GO TO 999←----- Insert text after pending line.
/C←----- Advance pending line.
      GO TO 999
      IF(IERR.EQ.3.OR.IERR.EQ.4) GO TO 20
/C 50←----- Use character replace to insert statement number
      50 IF(IERR.EQ.3.OR.IERR.EQ.4) GO TO 20 in pending line and advance pending line.
      WRITE(LUTTY,106)FNAME
/51
      WRITE(LULST,104)LINE,(INBUF(K),K=1,LEN)
/P 40←----- Use character replace to insert statement number
      40 WRITE(LULST,104)LINE,(INBUF(K),K=1,LEN) in pending line and keep pending line.
/60
      104 FORMAT(X,14,X,40A2
/GA2/A2)
      104 FORMAT(X,14,X,40A2)←----- Exchange to add right half or parenthesis ("") at
                                          end of line.
ELR&FLIST←----- Replace old file with edited text.
  LS FILE 2 28
END OF EDIT

```

## COMPILER LISTING OF &amp;FLIST

PAGE 0001

FTN4 COMPILER: HP24177 (SEPT. 1974)

```

0001 FTN4,L,B
0002 C
0003 C
0004 C
0005 C THE PROGRAM FLIST READS A TYPE 3 OR 4 FILE MANAGER FILE
0006 C AND PERFORMS A LIST FUNCTION (SAME AS '!LI,NAMR') TO THE
0007 C LINE PRINTER WITH LINE NUMBERS IN THE FIRST FIVE COLUMNS.
0008 C
0009 C THE SEQUENCE OF OPERATIONS FOLLOWS:
0010 C
0011 C (1) GET PARAMETERS AND SET UP DEVICES,DEFAULTING
0012 C IF NECESSARY.
0013 C (2) GET FILE NAME FROM USER AND OPEN IT.
0014 C (3) IF FILE ERROR,INFORM USER AND QUIT.
0015 C (4) IF FILE TYPE NOT 3 OR 4, INFORM USER AND QUIT.
0016 C (5) INITIALIZE LINE COUNT.
0017 C (6) READ A LINE FROM THE FILE.
0018 C (7) IF END OF FILE,WRITE MESSAGE AND QUIT.
0019 C (8) IF READ ERROR,WRITE MESSAGE AND QUIT.
0020 C (9) LIST THE LINE ON THE LIST DEVICE.
0021 C (10) UPDATE THE LINE COUNT AND GO BACK TO STEP (5).
0022 C
0023 C THE PROGRAM STARTS HERE.
0024 C
0025 C PROGRAM FLIST(3,90)
0026 C
0027 C INTEGER DCB(144),LUN(5),INBUF(40),FNAME(3)
0028 C
0029 C CALL RMPAR(LUN)
0030 C LUTTY=1
0031 C LULST=6
0032 C IF(LUN(1).NE.0) LUTTY=LUN(1)
0033 C IF(LUN(2).NE.0) LULST=LUN(2)
0034 10 FNAME(1)=2H
0035 C FNAME(2)=2H
0036 C FNAME(3)=2H
0037 C WRITE(LUTTY,100)
0038 C READ(LUTTY,101)FNAME
0039 C CALL OPEN(DCB,IERR,FNAME,0)
0040 C IF(IERR.GE.0) GO TO 50
0041 C WRITE(LUTTY,102)FNAME,IERR
0042 C GO TO 999
0043 50 IF(IERR.EQ.3.OR.IERR.EQ.4) GO TO 20
0044 C WRITE(LUTTY,106)FNAME
0045 C GO TO 999
0046 20 LINE=1
0047 30 CALL READF(DCB,IERR,INBUF,40,LEN)
0048 C IF(LEN.EQ.-1) GO TO 70
0049 C IF(IERR.EQ.0) GO TO 40
0050 C WRITE(LUTTY,103)FNAME,IERR
0051 C GO TO 999
0052 40 WRITE(LULST,104)LINE,(INBUF(K),K=1,LEN)

```

COMPILER LISTING OF &FLIST (Continued)

```
0053         LINE=LINE+1
0054         GO TO 30
0055     70 WRITE(LULST,105)
0056 C

0057     100 FORMAT(// " ENTER FILE NAME: ")
0058     101 FORMAT(3A2)
0059     102 FORMAT("FILE  '3A2'  ERROR:"17/)
0060     103 FORMAT("FILE  '3A2'  READ ERROR:"17/)
0061     104 FORMAT(X,14,X,40A2)
0062     105 FORMAT(// " END-OF-FILE"/"1")
0063     106 FORMAT("FILE  '3A2'  OF WRONG TYPE")
0064 C
0065     999 END
```

```
** NO ERRORS**      PROGRAM = 00492      COMMON = 00000
```

- A (*see* abort)
- abort, 3-1, 4-39
- advance pending line after edit, 2-3, 2-4, 4-20, A-3
- advance specified number of lines, 4-13
- ALGOL editing restrictions, 4-5
- assign edited text to LS area (*see* EL command)
- assign text to file, 1-7
- automatic text alignment, 1-4
  
- B command, 2-5, 4-26, 4-36
- back slash (*see* rubout)
- back up in destination work area, 4-16, 4-18
- backspace, 1-3
- batch mode, 1-1, 1-4, 5-1
- bell control, 4-4, 4-5
- blanks as first two characters of displayed line, 4-11
- brackets, 1-3
  
- C command to advance pending line, 2-3, 2-4, 4-20, 4-21, 4-22, 4-23, 4-24
- C control character
  - command, 4-23, 4-25
  - example, 2-3, 2-4
- cancel characters (*see* C control character)
- carriage return, 1-3, 2-3, 2-4, 4-21
- cartridge reference, 1-5, 4-41, 4-43, 4-44
- change EDITR prompt character, 1-5, 4-4, 4-5
- change tab character, 4-5, A-2
- character alignment, 4-20
- character edit
  - and pending line, 2-1
  - combinations, 4-24
  - commands, 4-20 thru 4-24
  - examples, 2-3, 2-4
  - summary, 4-24
- character field in sequence numbers, 4-6, 4-7
- character replace on pending line (*see* G command)
- clear LS area, 1-6
- column boundaries (*see* W command)
- command
  - input, 1-4
  - summary, 4-1
  - syntax, 1-3
- control character
  - and character delete, 2-4, 4-23, 4-24
  - and character insert, 2-3, 4-22, 4-24
  - and character replace, 2-4, 4-21, 4-24
  - and character truncate, 2-3, 4-23
  - symbol in examples, 1-2
  - syntax, 1-3
- control functions, 4-4
- copy pending line, 4-20, 4-31
- corrupt file, 3-1
  
- create
  - EDITR source file, 1-6
  - file manager file, 4-41
- CTRL/G, 4-4
- CTRL@, 4-25
- current delimiter
  - and pattern edits, 4-25, 4-34
  - in character edits, 2-3
  - position to character edits, 2-4
  - used to preserve existing text, 4-20
- current line available to edit (*see* pending line)
- current line edits, 2-3
  
- D command, 4-28 thru 4-30, 4-36
- default line length, 1-4
- default mode edit, 4-21
- delete
  - characters, 2-4, 4-23, 4-37
  - destination file, 3-1
  - lines of text, 1-3, 4-19, 4-31
  - pending line, 2-1
- delete character, 1-3
- delete lines of text to a find field (*see* D command)
- deletions, 1-2
- destination work area, 1-2, 2-1, 4-14, 4-16
- display
  - a number of lines, 4-11, 4-13
  - a specified line, 4-12
  - approximate number of words in destination work area, 4-16
  - commands, 4-11 thru 4-16
  - destination work area, 4-16
  - line number in destination work area, 4-14
  - number of characters in pending line, 4-15, 4-17
  - pending line, 4-11, 4-13
  - pending line number, 4-14
  - specified line, 4-14
- displayed line, 1-2
  
- EC command, 4-41
- edit existing file, 1-6
- edit preceding line, 1-1
- EDITR
  - abort, 3-1, 4-39
  - and spooling, 1-4, 5-1
  - batch mode, 1-1, 1-4, 5-1
  - display format, 1-4
  - enter commands, 1-4
  - interactive mode, 1-1, 1-5
  - loading, 1-7/1-8
  - record size, 3-1, 4-15
  - work areas, 1-1, 1-2, 2-1, 4-14, 4-16
- EL command, 4-40
- ELC command, 4-43



ELR command, 4-44  
empty LS area, 1-6  
enable exchange pattern all lines (*see* X and Z commands)  
end edit  
    and create file manager file (*see* EC, ELC commands)  
    and replace old file manager file (*see* ER, ELR commands)  
    and store destination file in LS area (*see* EL command)  
end-of-file, 3-1, 4-26, 4-27, 4-36  
ER command, 4-42  
erase lines in destination work area, 4-16  
error message  
    display format, 1-4  
    summary, 3-1  
ESC, 1-3, 4-25, 4-27, 4-28, 4-29, 4-30, 4-31  
escape character (*see* ESC)  
escape key, 1-3  
exchange commands  
    described, 4-24, 4-33 thru 4-38  
    example, 2-5  
exchange on pending line (*see* Y command)

F command, 2-5, 4-10, 4-27 thru 4-28, 4-35  
file manager commands, 1-4, 5-1, 6-1  
file name  
    format, 1-5  
    larger than line length, 4-7  
    number of characters, 1-6  
find a line with a find field  
    SOF to EOF, 4-26  
find a line with a find field from  
    pending line to EOF, 4-27  
find field  
    exchange commands, 4-33  
    search commands, 4-25  
FMGR 006, 1-6

G command, 4-33, 4-34, A-2, A-3, A-4

H command, 4-15, 4-17

I command, 2-1, 4-18, 4-20  
incrementing value in sequence numbers, 4-6, 4-7  
indefinite length field, 1-3, 4-25  
initial tab stops, 4-5  
initial window field, 4-5  
insert  
    a letter in a word, 4-22  
    characters, 2-3, 4-22, 4-24, 4-37  
    text after the pending line, 1-3, 2-1, 4-18, 4-21  
    text before pending line, 2-2, 4-18, 4-20  
    zero length line, 4-18  
insertions, 1-2  
interactive mode, 1-1, 1-5  
    prompt character, 1-5

J command, 4-31 thru 4-33  
job command file, 5-1  
jump to a line with a find field (*see* J command)

K command, 4-8, 4-10  
keyboard commands, 1-2  
keyboard conventions, 1-3  
kill trailing blanks (*see* K command)

limit area of record scanned in search or exchange, 4-5  
line edits, 1-2, 2-1, 4-17 thru 4-19  
line length, 1-4, 4-7, 4-13  
literals, 1-3, 4-33  
LN, 4-13  
loading EDITR on-line, 1-7/1-8  
logical unit  
    negative, 4-41  
    of command input to EDITR, 1-4  
    of spool file, 5-1  
lower case italics, 1-3  
LS tracks, 1-3, 1-6, 4-40, 4-42, 4-43, 6-1  
LU (*see* logical unit)

M command, 4-9 thru 4-10, A-2  
maximum output line length, 1-4  
merge source file (*see* M command)  
move lines of code, 4-31  
multi terminal monitor, 1-4, 6-1

N command  
    to display pending line number, 4-14, 4-16  
    to display specified line, 4-12  
name, 4-41, 4-42  
NAMR, 1-5  
ND command, 4-14, 4-17  
new file name, 4-42, 4-43, 4-44  
normal terminate sequence, 5-1

O command, 4-20, 4-21, 4-22, 4-23, 7-1  
ON command, 1-4  
output file, 1-1

P command  
    display pending line, 4-11, 4-15  
    retain pending line after edit, 2-3, 2-4, 4-20, 4-21, 4-22, 4-23  
paper tape breaks, 4-16  
pattern edit  
    combinations, 2-5, 4-35  
    commands, 4-24 thru 4-38  
    example, 2-5  
    initial pointer position, 2-5  
pending line  
    and pattern edits, 4-25, 4-28, 4-31, 4-36  
    line pointer, 2-1  
    number, 4-14  
    number of characters, 4-15, 4-17  
    zero length, 4-17  
program copy directions, 6-1  
prompt character, 1-5

Q command, 7-1

R command  
    command description, 4-17  
    define replace field, 2-4  
    line edit function, 2-1  
record longer than 150 characters, 3-1

- replace a line of text, 2-1
- replace characters
  - described, 4-21, 4-37
  - example, 2-4, A-2
- replace file manager file (*see* ER command)
- replace pending line with text, 4-17, 4-19
- reset window boundaries, 4-5
- return, 1-3
- RN command, 6-1
- RP command, 6-1
- RU command, 1-4
- rubout, 1-3

- S command, 4-16
- search commands
  - defined, 4-24, 4-25
  - example, 2-5
- security code, 1-5, 4-41, 4-43, 4-44
- semicolon (;), 4-5, 4-25
- sequence identifier, 4-6
- sequence numbers
  - column numbers specified, 4-6
  - output on line printer, 4-7
- set line length, 4-7
- set tab stops, 4-5, 4-6
- set window, 4-5, 4-6, 4-34, 4-35, A-3
- slash (/)
  - advance pending line, 4-13
  - as prompt character, 1-5
  - current delimiter, 4-4
  - in batch mode, 5-1
  - in examples, 4-1
  - in find fields, 4-25, 4-26
  - place holder in character edits, 2-3, 4-21, 4-22, 4-24
  - request input, 4-4
- source file, 1-1, 1-6, 4-39, 4-42
- source work area, 1-1, 1-2, 2-1
- SP command, 6-1
- space ( $\Delta$ )
  - as a place holder to cancel characters, 4-24
  - to insert after a line (of text), 1-3, 2-1, 2-2, 4-18, A-2
  - to position characters within a line, 4-8
  - to request LS area, 4-40, 4-41
- space down a number of lines, 4-13
- spool file, 1-34, 5-1
- starting sequence number, 4-6, 4-7
- syntax error, 3-1

- T command, 4-5
- tab, 4-5, 4-25, 4-34, 7-7, A-2
  - and cancel characters, 4-23
- terminate
  - commands, 1-7, 3-1, 4-39 thru 4-44
  - errors, 3-1, 4-39
- text
  - creation, 1-6
  - display format, 1-4
  - edit command summary, 4-1
  - storage, 1-7
- TIME program listing, 4-1
- transfer file, 6-1
- truncate
  - characters from pending line, 2-3
  - result of maximum line length, 4-7
- type 0 file, 1-6, 1-7, 4-39

- U command, 4-33, 4-38
- unconditional character replace (list)-*see* V command
- unconditional character replace (no list)-*see* U command
- upper case block letters, 1-3
- upper case italics, 1-3

- V command, 4-33, 4-37
- variables, 1-3

- W command, 4-5, 4-6, 4-34, 4-35, 4-37, A-3
- window (*see* W command)

- X command
  - and find field, 4-33
  - and window, 4-5
  - as pattern edit, 4-36
  - change prompt, 4-4, 4-33
  - example, 2-5
  - reenable exchange, 4-36

- Y command, 4-33, 4-35

- Z command, 4-33, 4-36, 4-38
- zero length line, 4-17, 4-25, 4-28, 4-34, 4-35, 4-36



## READER COMMENT SHEET

RTE Interactive Editor  
Reference Manual

92060-90014

June 1978

We welcome your evaluation of this manual. Your comments and suggestions help us improve our publications. Please use additional pages if necessary.

**Is this manual technically accurate?**

**Is this manual complete?**

**Is this manual easy to read and use?**

**Other comments?**

---

**FROM:**

**Name** \_\_\_\_\_

**Company** \_\_\_\_\_

**Address** \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

FOLD

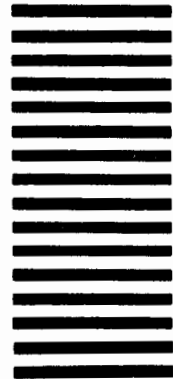
FOLD

**BUSINESS REPLY MAIL**

No Postage Necessary if Mailed in the United States Postage will be paid by

Hewlett-Packard Company  
Data Systems Division  
11000 Wolfe Road  
Cupertino, California 95014  
**ATTN: Technical Marketing Dept.**

FIRST CLASS  
PERMIT NO. 141  
CUPERTINO  
CALIFORNIA



FOLD

FOLD

PART NO. 92060-90014  
Rev. Code 1805  
Printed in U.S.A. 5/78



Sales and service from 172 offices in 65 countries.  
11000 Wolfe Road, Cupertino, California 95014