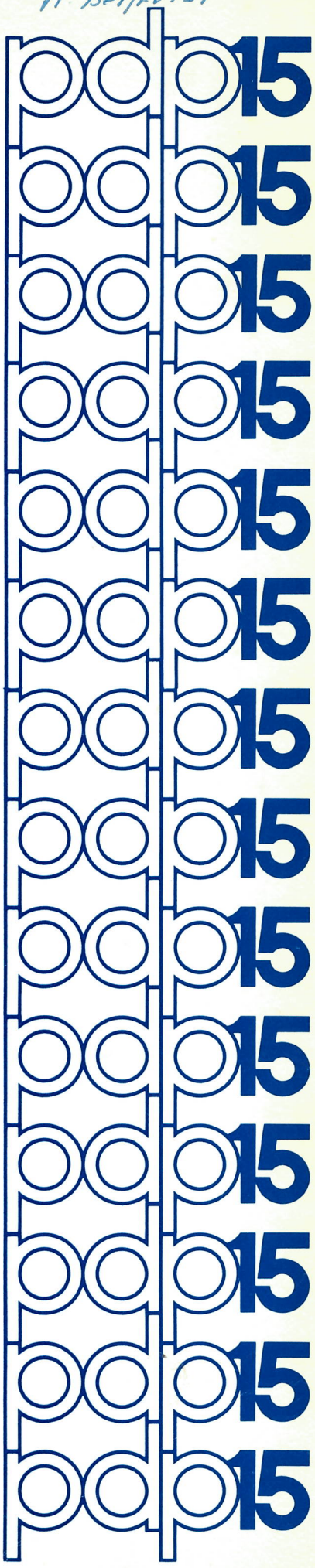
H. Berglund

digital

edit

utility program



digital equipment corporation

DEC-15-YWZB-DN6

P D P - 1 5

T E X T E D I T O R

Utility Program

For additional copies of this manual order No. DEC-15-YWZB-DN6
from the Program Library, Digital Equipment Corporation,
Maynard, Massachusetts 01754

First Printing
October 1971

The information presented in this manual is for informational purposes only and is subject to change without notice.

PREFACE

The operation and use of the PDP-15 Utility Program EDIT is described in this manual. The information presented is valid for use in either the ADVANCED Software System (ADSS) or the Disk Operating System (DOS) environments. Differences between editing in ADSS and in DOS environments are indicated.

It was assumed in the preparation of this manual that the reader is familiar with the operation of the PDP-15 equipment and the contents of the software manual describing the features of the particular monitor system in use, that is:

a) for ADS users, PDP-15/20/30/40 ADVANCED Monitor Software System manual, DEC-15-MR2B-D;

b) for DOS users, DOS Software System Users Manual, DEC-15-MRDA-D.

PDP-15 UTILITY PROGRAMS MANUAL, DEC-15-YWZB-D

The PDP-15 Utility Programs manual is comprised of a set of individual manuals, each of which describes the operation and use of a PDP-15 Utility Program. The manuals which make up the Utility Programs set are listed in the following Application Guide. In addition, the Application guide also indicates the order number of each manual and the specific PDP-15 Monitor Software Systems in which the program described may be used.

The Utility Manuals may be ordered either individually, by using the title and order number given with each mauual, or as a set, by referencing "PDP-15 Utility Programs Manual, DEC-15-YWZB-D".

CONTENTS

EDITOR

CHAPTER 1

EDITOR

SECTION 1

INTRODUCTION

The Text Editor (EDIT) is a powerful context-editing program that allows the modification and creation of symbolic source programs and other ASCII text material[1]. By means of keyboard commands, the Editor is directed to bring a line, or group of lines, from the input file to an internal buffer. The user can then, by means of additional commands, examine, delete, and change the contents of the buffer, and insert new text at any point in the buffer. When the line or block of lines has been edited, it is written into a new file on the output device.

The Editor is most frequently used to modify MACRO and FORTRAN IV source programs, but it can also be used to edit any symbolic text.

The EDIT program operates in either the ADVANCED Monitor (ADSS) or the Disk Operating (DOS) Software System. Two versions of the EDIT program designed to use available CRT display systems for soft-copy display purposes during EDIT operations are described in Appendices B and C.

Appendix B contains a description of the use of EDITVP, an editor program which utilizes the VP15A Storage Tube Display unit as the editing data display device instead of the console printer. This editor program may be used in both the ADVANCED and DOS software systems.

Appendix C contains a description of the use of EDITVT, an editor program which utilizes the VT15 Graphic Display Unit as the editing display device instead of the console printer. This editor program may be used only in DOS software systems.

A summary of the standard Editor commands is presented in Appendix A.

---

[1]The Editor reads and writes standard IOPS ASCII lines. The characteristics of IOPS ASCII text are described in the ADVANCED Monitor manual for the PDP-15/20/30/40 Systems Software (ADSS) (DEC-15-MR2B-D), and in the DOS Software System Users Manual (DEC-15-MRDA-D)

SECTION 2

FUNCTIONAL DESCRIPTION

2.1 CONTROL MODES

The Editor operates in either an EDIT or INPUT control mode. In the EDIT (or Command) Mode, the program accepts and acts upon control word and data strings to open and close files; to bring lines of text from an open file into the work area; to change, delete, or replace the line currently in the work area; and to insert single or multiple lines after the line in the work area. In Input (or Text) Mode, lines from the keyboard are interpreted as text to be added to the open file. Commands are available for conveniently changing control mode.

2.2 DATA MODES

Data from the input file is made available for editing in two ways: in Line-by-Line Mode or in Block Mode.

The Line by Line data mode is the program "default" mode if both the input and output devices are directoried devices. If either or both of the devices involved are non-directoried, the program default is the Block Mode.

2.2.1 Line-by-Line Data Mode

In Line-by-Line Data Mode, a single line is the unit of the input file available to the user for modification at any point. The line currently available is specified by a pointer, which can be thought of as moving sequentially through the file, starting just before the first file line, in response to typed editing commands. When a file is opened at the be beginning of an editing session, the first line of that file can be brought into the work area and is available for modification. This line remains in the work area until the user requests that a new line be brought in. The pointer then moves down the file until the line requested is encountered. That line is brought to the work area and, as the "current" line, can be modified. Lines previously skipped over are no longer available for editing by the user, but are written in the output file. Thus, at any point in a single edit run in Line-by-Line mode, the user is able to modify only the portion of the input file consisting of the current line and all lines between the

current line and the end of the file (i.e., the current line and all lines below it).

2.2.2 Block Data Mode

In Block Data Mode, a user-specified portion[1] of the input file is held in a core buffer for editing until the user requests that the contents of the buffer be added to the output file. A group of Editor commands is available for use in Block Mode only (see Section 4) in addition to the commands used in line-by-line editing.

When the user is operating in Block Mode, commands to the Editor are honored only with respect to that portion of the input file currently occupying the buffer. The lines of text in the buffer are made available for modification through the use of normal locative requests and can be reaccessed until the buffer is emptied by the user.

Unless deleted, lines passed over in Block Mode are available to the user until the contents of the buffer are written in the output file. Consider, for example, the editing request to search for and bring in a specified line. In Line-by-Line Mode, the result is a scan of (possibly) the entire file below the pointer. The same request in Block Mode provides a search of the entire buffer below the pointer, but no further.

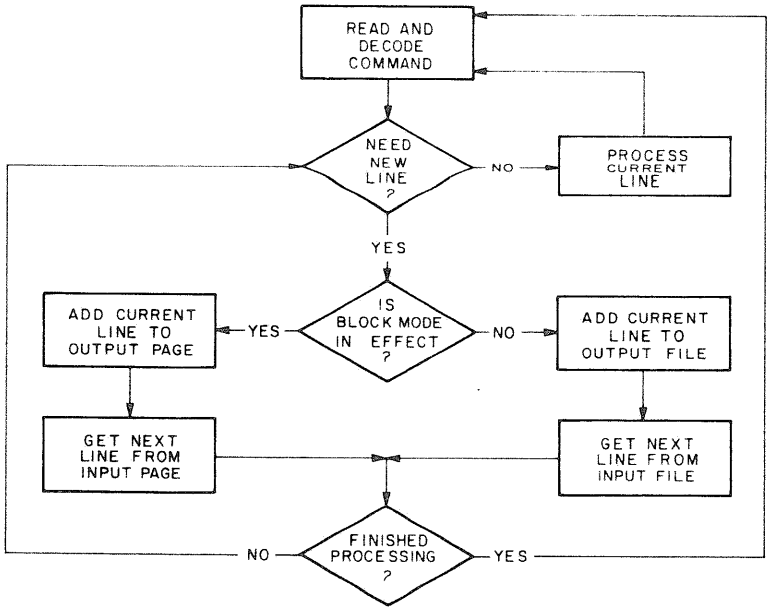


Figure 2-1. Schematic of Line Processing in Block and Normal Modes

[1] If not specified, a group of 55_{10} lines is assumed (default size) by the Editor.

Block Mode has another advantage: rapid correction of editing command errors. If the user finds that he has typed the wrong command, he can immediately correct it, because the buffer has not been added to the output file. In Line-By-Line Mode, a command error may cause the program to bypass a line in which a change is needed. The user must then output a new input file and begin editing again (i.e. via the TOP(T) command).

2.3 DATA FILES

The following paragraphs describe the Monitor I/O features utilized by the Editor program, how data may be added to files and the operations performed on the Editor output files.

2.3.1 Using Monitor I/O

In both DOS and ADSS software environments, the Editor makes use of the monitors' Input/Output Programming System (IOPS) for I/O transfer operations. In DOS systems, the Editor communicates with IOPS via separate Device Assignment (DAT) and User File Directory (UFDT) tables. Refer to the DOS User's manual (DEC-15-MRDA-D) for a description of these tables and their use.

TABLE 2-1

Standard DAT and UFDT Slot Assignments for ADSS and DOS Monitors

ADVANCED MONITOR .DAT Slot Assignments

.DAT Entry Number	Used for
-3	Teleprinter output; messages to user
-2	Keyboard input; text and commands
-14	File input
-15	Scratch or edit file output
-10	Subsidiary input
+10	VP 15A (EDITVP only)

DOS MONITOR .DAT Slot Assignments

.DAT	DEVICE	UIC[1]	USE
-15	DKA	GEP[1]	Output/Scratch
-14	DKA	GEP	I/O
-10	TTA	GEP	Secondary Input
+10	VPA	GEP	(EDITVP only)

[1]The current UIC will be given here, GEP is an example.

In ADSS systems communications with IOPS is carried out via a DAT table (refer to ADSS Monitor manual DEC-15-MR2B-D).

In both software systems the tables involved are provided with a set of standard "slot" device and UIC assignments (shown in Table 2-1); these assignments may be altered for an editing operation by using the monitor ASSIGN command, prior to loading the Editor program, to change the table assignments as desired (refer to the respective monitor or keyboard guide for procedure - see Table 1-1).

2.3.2 Input and Subsidiary Files

The Editor accepts inputs from:

a) the console keyboard (i.e. commands and text entries),
b) a file input device/UFD which contains the file to be edited,
c) a subsidiary input device/UFD which contains, normally, prepared material to be inserted into the object file.

If the file input and subsidiary inputs are ignored (not assigned) by the user, the Editor assumes that all inputs will come from the keyboard.

When used, the subsidiary input device must contain non-file structured data, normally the paper tape and card reader devices are used. However, DECtape or disk devices may be assigned as subsidiary devices if they contain information written in a non-file structured manner (refer to appropriate monitor manual). If a device which normally contains file structured data is specified as the subsidiary device, the Editor outputs the following message:

```
.DAT-10 IS FILE ORIENTED
DO YOU WISH TO CONTINUE ?
```

on the console printer.

The user then either:

a) types the word YES to continue the operation, or
b) enters any other word or character to return operations to the monitor where .DAT-10 may be reassigned.

2.3.3 Output Files

The Editor attempts to determine whether the input and scratch devices are file-structured immediately on receiving control after having been loaded. If either one of the devices is not file-structured, then the scratch device (DAT entry -15) is assigned as the final output device. If both devices are file-structured, the scratch device is assigned an intermediate function and the input device is used as the final output device.

If, in DOS editing operations, the same device and UIC are specified as both the input and output devices, then the file CLOSE operation will cause the edited file to be renamed, not recopied.

The intent, in all cases, is to allow replacement of the input file by the edited output file. This is possible only when the input and output devices can be both read and written. If replacement can be accomplished (both devices are file-structured), the following sequence of events takes place when the files are closed after editing.

a. The intermediate output file is read from the scratch device and written on the input device under a temporary name.
b. The old input file is deleted from the input device.
c. The intermediate output file is deleted from the scratch device.
d. The intermediate output file, temporarily named and now residing on the input device, is given the name previously assigned to the old (now deleted) input file.
e. The output file is closed and immediately becomes available for use.

If no replacement can be accomplished, no change is ever made to the input file. If the output device is file-oriented, the new edited file is properly entered in the file directory for that device under the name given in the OPEN, CLOSE, or EXIT command sequences.

The possible destinations of the new edited file are summarized in Table 2-2.

Note that in the process of file housekeeping, there is always at least one copy of the output file available on one, or both, of the devices. Further, the original input file is not deleted until the new file has been successfully written and closed. A system failure,

therefore, can never result in total loss of data. Recovery procedures to be used in case of difficulty are outlined in Chapter 5.

TABLE 2-2

Output File Conventions for the Text Editor

Input Device	Scratch or Output Device	Edited File Appears as:	Input File is:
File oriented	File oriented	Input Device	Deleted
File oriented	Non-file oriented	Output Device	Unchanged
Non-file oriented	File oriented	Output Device	Unchanged
Non-file oriented	Non-file oriented	Output Device	Unchanged

2.4 USE OF THE CTRL P KEYBOARD COMMAND

The command CTRL P when:

a) issued during the entry of a command string (before terminator) causes the Editor to be restarted.

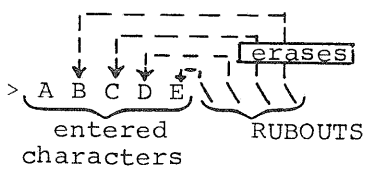
b) issued during the process initiated by a command string, aborts the process on completion of the current logical sequence (ignored during MOVE command operations);

c) issued during an INPUT mode, causes all data entered on the current line to be deleted and switches the mode to EDIT. A second CTRL P entry at this time would restart the Editor.

2.5 CHARACTER ERASE AND LINE DELETE KEYBOARD COMMANDS

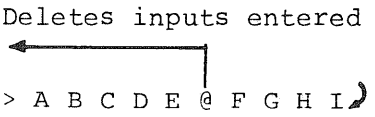
Two keyboard commands are provided the user which permit the deletion (erasure) of individual characters of a line and the deletion of a line or line segment. The command:

a) RUBOUT, permits the deletion of individual characters. Each operation of the keyboard RUBOUT key deletes the last entered (non-erased) character of the current line. The symbol (\) is echoed on the console printer each time RUBOUT is used. The following illustrates the RUBOUT operation:



results in the line >A

b) CTRL U causes the deletion from the current line of all characters entered before the CTRL U was issued. Each CTRL U entry causes the symbol (@) to be echoed at the console printer. The following illustrates the CTRL U operation:



results in a final line of

> F G H I

NOTE

In addition to its editorial function, CTRL U may also be used to terminate Editor output operations. Entered at any time during an output operation, CTRL U immediately terminates the operation and initiates a carriage/return.

During editorial operations, neither the RUBOUT nor CTRL U command affects previously prepared text.

SECTION 3

EDITING OPERATIONS

The Editor always begins in the EDIT mode and assumes that the user wishes to modify some (named or unnamed) file. When first loaded or restarted, the EDITOR program outputs the message:

```
EDITOR Vnnx
```

where: nn = a one- or two-digit version number

at the console printer and waits for the user's first command.

3.1 MODIFYING AN EXISTING FILE

If the input device is file-structured (disk, drum, magnetic tape, or DECtape), the first command to the Editor must be:

```
OPEN filename ext
```

where "filename" is the primary name[1] of the wanted file residing on the input device and "ext" is its extension.[2] Ext can be omitted and, if so, is assumed to be SRC. If the file specified is not found in the directory, the program assumes that the user wishes to create a file named "filename ext". Accordingly, when it has been determined that the named file is absent from the input device, the Editor types:

```
FILE filename ext NOT FOUND.
INPUT
```

INPUT mode is entered and subsequent lines from the keyboard are inserted in a new temporarily named file on the output device.

If the specified file is present on the input device, an intermediate temporarily named file is opened for writing on the output device, and the input file is opened for reading. The user may then proceed to make the necessary changes in the input file.

---

[1]Maximum of six characters permitted for "filename".

[2]Maximum of three characters permitted for "ext".

If the input device is not file-structured (e.g., paper tape reader, card reader), the user's first command after program initialization can be any edit request. The OPEN command is not required for nonfile-structured devices.

3.2 INPUT/EDIT MODES

To enter text from the keyboard, the Editor must be in INPUT Mode. To carry out an edit function on the current line, the Editor must be in EDIT Mode.

Control may be changed at any time with a line of zero length (typing a carriage return only). The Editor command INSERT (without arguments) also causes a mode change. After the user changes control modes, the Editor types INPUT or EDIT, indicating the control mode in effect.

3.3 CREATING A NEW FILE

When both input and output devices are file-structured, the user can issue in EDIT mode the OPEN command, followed by the name he wishes to assign to his new file. Since a file of the name given is guaranteed not to be found (if the user has properly chosen his new name), INPUT Mode is immediately entered, following the standard error message. The name specified is assigned to the final output file if no other name is given in the CLOSE command.

If the output device is file-structured, a temporarily named file is opened for writing; input lines from the keyboard are added to it as they are entered. If the output device is not file-structured, the file-naming conventions may be bypassed.

3.4 EDIT BLOCK MODE

If the BLOCK mode is not the current Editor default mode (refer to paragraph 2.2), the user must enter the keyboard command BLOCK ON to establish this data mode.

During Block mode edit operations, the input file is edited in multi-line groups rather than line-by line. Each group of lines is loaded (read) into a Block buffer where edit operations are performed. On completion of the edit operations, the contents of the block buffer are written into an output file. The number of lines within each "buffer

group" may be specified by the user by means of a "SIZE" command (see paragraph 4.22). If not specified, the Editor assumes a default size of 55_{10} lines.

To return to a line-by-line Edit Mode, the user must issue a "BLOCK OFF" command. If the BLOCK OFF command is issued before the Block Buffer is empty, the Editor outputs the message:

BUFFER NON-EMPTY

at the console printer. The user must then empty the buffer before the Block Edit Mode can be terminated. Block buffers are emptied using the command WRITE (refer to 4.3.2).

3.5 CLOSING THE NEW FILE

When the user, after modifying or entering an input file, is satisfied that all needed operations have been carried out, he must close out the input and output files using the edit command:

CLOSE␣␣filename␣␣ext)

which initiates the sequence of events described in paragraph 2.3.3. Neither "filename" nor "ext" need be specified if previously given in the OPEN command. Any filename and ext given in the CLOSE command string override the names given in the OPEN command.

3.6 ERROR-HANDLING CONVENTIONS

3.6.1 Command String Errors

All mistakes in the use of Edit Mode control words result in a common complaint by the Editor. Although the possible errors in usage fall into a number of distinct categories, the program makes no attempt to differentiate among error types. The reasons for this common treatment lie in the requirement that the Editor take some cognizance of its memory allocation (relatively obscure error types need as much memory for recognition and response as do the more usual mistakes) and in the fact that the treatment rendered makes the error self-explanatory.

Command string errors, then, all result in the single typed comment,

NOT A REQUEST:

followed, on the next line, by the request line with which the Editor had trouble.

Usual types of command string errors include the following:

a. The edit control word issued was not among those in the program's repertoire.
b. A SIZE command was issued with a missing argument or an argument of "1".

When the BRIEF Mode (see Paragraph 4.2.7) is ON, the Editor command and the command line in error are replaced by a single typed question mark:

```
?
```

3.6.2 Premature End-of-File

During the processing of some commands, it occasionally happens that a read is attempted which moves the pointer below the last line of a logical (or physical) group. Consider, for example, the effect of a numeric argument in the GET n command line. The program reads successive lines from the subsidiary input device until exactly n lines have been read. If, in the process of reading, it is discovered that fewer than n lines are physically present on the secondary input medium (paper tape, say), then a premature end-of-file condition is said to exist. An improperly formulated FIND request (the character string typed is absent from the file) results in a similar condition.

Depending on the character of the incoming group of lines (block buffer, secondary input medium, or input file), the appearance of an unexpected end-of-file causes a comment to be typed to inform the user of the difficulty. The form of the message is:

END OF {BUFFER / FILE} REACHED BY:

followed, on the next line, by the edit request which caused the problem.

A premature end-of-file causes the pointer to be left <u>below</u> the final line of the group being read.

3.6.3 Read Errors and Line Overflow

During DOS editing operations, inputs from either the input or subsidiary file are checked for two possible errors (in ADSS operations, only inputs from the input file are checked. Whenever a line containing an error is found, it is printed at the console and made the "current line" to enable it to be modified by the user.

The first type of error occurs when the input file device handler detects either incorrect parity or a faulty checksum in the incoming line. The printed comment is:

READ ERROR:

followed by the line in which the error was encountered.

The second difficulty results from the appearance of a line which is too long to be contained in the program's internal buffers. Any line of more than 89_{10}[1] characters plus a terminator results in the comment:

TRUNCATED:

followed by the first (leftmost) 89 characters of the long line. The remaining right-end characters are discarded. The user has the choice, after either type of error, of modifying the line which causes the complaint (via any manipulative request) or of allowing the line to stand as is in the output file (via any locative request).

3.6.4 Block Mode Buffer Overflow

When Block Mode is in effect, it is possible for an attempted addition of a line to the Block Mode Buffer to exceed the buffer's capacity. This might occur, for example, during the processing of a READ request if the number of input lines (previously defined by a SIZE command) is too great to be accommodated by the memory available. When the capacity of the buffer is exceeded, the program types the comment:

BUFFER CAPACITY EXCEEDED BY:
(offending line)

---

[1] A 90_{10}-character line is permitted in the ADSS Monitor System.

To eliminate this error condition, the operator must delete the excess (offending) line. The user should control carefully the size specification of the buffer and lines entered to ensure that this error condition is avoided.

3.6.5 File Naming and Calling Errors

Errors in file name usage can be classified in three general groups: 1) either the named file cannot be found, 2) a name has not been given to the file at a point where one is needed, or 3) a name has been given which cannot be used.

3.6.5.1 Absent File - If the file named in the OPEN request line cannot be found on the device associated with .DAT slot -14, the assumption is made that the user wishes to create a new file with the name given. The program prints the comment:

FILE [filename ext] NOT FOUND.

and changes to Input Mode.

3.6.5.2 Absent File Name - If no file name is given either in an OPEN request line or as an argument to the CLOSE command, the program, after attempting to process the CLOSE request, prints:

NO FILE NAME GIVEN.

The next edit request must be another CLOSE naming the file.

If no OPEN command is issued (a new file is being created), any locative request (FIND, NEXT) results in the comment:

NO INPUT FILE PRESENT.

3.6.5.3 Identically Named Files - The problem of duplicate file names is apparent on two levels. In the first case, it is possible for a previous edit run to have been aborted with one of the Editor's temporary files (normally .TFIL1 EDT) closed on the output device. The closing of the temporary file created during the current edit run results in the deletion of the line-named file from the previous run. To enable the retrieval of prior work, the Editor types the comments:

```
FILE .TFIL1 EDT IS PRESENT ON OUTPUT DEVICE.
PLEASE RENAME IT OR IT WILL BE DELETED.
```

If the user wishes to preserve the contents of .TFIL1 EDT, he must rename it using the CALL request (see Paragraph 3.7).

At the second level, it may happen that the file name given in a CLOSE sequence is identical to that of another file on the (current) output device. In such cases, the program types:

a) in DOS systems, the message:

```
PLEASE TYPE IN ANOTHER NAME
```

and outputs a go-ahead symbol (>). The Editor then waits for the user to enter another file name and extension (SRC assumed if none given). The request message will be repeated until an acceptable file name is entered.

b) in ADSS systems, the message:

```
PLEASE USE ANOTHER NAME
```

A second CLOSE request (with a unique name) can then be given.

If file processing has proceeded to a point at which recovery, as described above, is impossible, the Editor recognizes a priority scheme when file name difficulties are encountered. An attempt is made, first, to ensure that the new (modified) version of the file being edited is left on .DAT slot -14 and properly named. If that is impossible, the program tries to leave the new file (again, properly named) on .DAT slot -15. If that cannot be done either, then the new file is left on .DAT slot -15 and is named .TFIL1 EDT. The Editor then reports the nature of the difficulty, the final destination of the file, and its current name, thus:

```
FILE [filename ext] IS PRESENT ON OUTPUT DEVICE
YOUR EDITED FILE IS ON .DAT-15 (OR -15) AS [newfile ext]
ORIGINAL FILE DELETED.
```

The user now knows the residence of his edited file (.DAT-14 or .DAT-15) and the name under which it can be accessed.

3.6.5.4 Nothing in File - The following error message can result from issuing CLOSE command prior to WRITE command when Block Mode is ON, or by having OUTPUT turned off when a WRITE or CLOSE command is issued:

NOTHING IN FILE

In any event, the control returns to the Editor. The contents of the buffer remain unchanged. In the case of file-oriented input and output, the input file is left unchanged.

3.7 FILE RENAMING AND DELETION

The Editor provides two commands:

a) CALL DELETE
b) CALL RENAME

which enable the user to either delete or rename any file on the Editor's input or output devices/UFD's.

These commands are described in detail in Section 4.

SECTION 4

EDIT MODE COMMANDS

4.1 INTRODUCTION

The form and use of each command permitted in Edit Mode are described in this section. The command descriptions are presented in five (5) basic functional groups (i.e., setup, I/O, locative, manipulative, and close); examples are given where possible. The following conventions are observed throughout this section:

a. Where commands may be abbreviated, the legal abbreviation is specified, within brackets, immediately after the command's full name. For example, "NEXT [N] - indicates that the command NEXT may be abbreviated as "N"

b. When arguments may be added to or are required as part of a command, the command name must be separated from its argument by a single blank character (space ␣). The blank delimiter is considered as part of the command, not part of the command's argument string. For example, the command RETYPE␣/COMMENT results in the line: /COMMENT.

If more than one blank appears between the command and its argument string, all blanks except the first are considered as part of the argument.

For example, the statement:

FIND␣␣␣/COMMENT)

results in a search for a line which begins with the character string

␣␣/COMMENT)

4.1.1 Example File

The following file or portions of it are used throughout this section to illustrate the operations performed by the Edit commands. The contents of this file (i.e., EXAMP) are meaningless; this enables the command functions to be demonstrated without regard to proper programming techniques or requirements. The file is:

```
          .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1      AAA     /LINE 1
          BBB
          CCC
TAG2      DDD     /LINE 4
          EEE
          FFF
TAG3      GGG     /LINE 7
          HHH
          III
          .END
```

4.2 SETUP COMMANDS

Edit Mode setup commands are required at the start of an editing session to define and give parameters for the possible Editor features which the user may employ and to delete or remove the file residing on the input device. The setup commands are described, individually, in paragraphs 4.2.1, through 4.2.9.

4.2.1 CALL RENAME Command

Form:

CALL␣RENAME␣{INPUT / OUTPUT}␣OLDNAM␣EXT␣NEWNAM␣EXT)

Description:
This command must be used before the "OPEN" command (see 4.2.3) is issued. It may be used repeatedly, and may be issued in any order with the "CALL DELETE" command (see 4.2.2).

The RENAME command permits the user to change the name of any file located on either the input or output device. No abbreviation is permitted.

Example:
The following example illustrates the use of the RENAME command in a DOS system to rename input file XXX SRC to the name NMTST SRC.

```
   02-OCT-71
DIRECTORY LISTING  (VIC)          listing of current UFD in
  2436 FREE BLKS                    DOS System
     1 USER FILES
     1 USER BLKS
XXX    SRC       1  02-OCT-71     entry for file "XXX SRC"
```

```
 EDITOR V18A
>CALL RENAME INPUT XXX SRC NMTST SRC   RENAME Command, change name
>                                        of file XXX SRC to NMTST SRC


    02-OCT-71                          listing of resultant UFD
DIRECTORY LISTING  (VIC)
  2436 FREE BLKS
     1 USER FILES
     1 USER BLKS
NMTST  SRC       1  02-OCT-71          entry of renamed file
```

4.2.2 CALL DELETE Command

Form:

CALL␣DELETE␣{INPUT / OUTPUT}␣FILNAM␣EXT

Description:

This command must be issued before the "OPEN" command (see 4.2.3) is issued. It may be used repeatedly and may be issued in any order with the "CALL RENAME" command (4.2.1). The DELETE command permits the user to delete any file from the directory of any device (or UFD) used as either the input or output device. No abbreviations are permitted.

Example:

The following example illustrates the use of the DELETE command in a DOS system to delete the file "NMTST SRC" from the current (input) UFD.

```
    02-OCT-71                          listing of current UFD in DOS
DIRECTORY LISTING  (VIC)                 System
  2436 FREE BLKS
     1 USER FILES
     1 USER BLKS
NMTST  SRC       1  02-OCT-71          entry for file "NMTST SRC"


 EDITOR V18A
>CALL DELETE INPUT NMTST SRC           DELETE Command, delete file NMTST
>                                      from input device (i.e., current
                                       UFD)
```

```
   02-OCT-71
DIRECTORY LISTING  (VIC)
 2437 FREE BLKS
    0 USER FILES
    0 USER BLKS
```

listing of resultant UFD

4.2.3 OPEN Command

Form

OPEN␣␣filename␣␣ext)

Description:
Edit Mode operations are initiated by the entry of this command. When entered, the Editor searches the Input device for the named file, if the file is found the program outputs the word "EDIT" and a go-ahead symbol (i.e., >); if the file is not found, the program outputs the message:

```
FILE NOT FOUND
INPUT
```

The word "INPUT" indicates that an intermediate write file has been opened on the output device. Any inputs from the Keyboard are written into the intermediate file, line-by-line.

Normal files found on the input device are open for reading and subsequent Edit operations.

Example:
The following are examples of the responses obtained when an existing and a non-existent file are named in an OPEN command.

```
 EDITOR V18A
>OPEN EDTST SRC
EDIT
>
```

opening an existing file

```
 ↑P
 EDITOR V18A
>OPEN AAA SRC
FILE AAA SRC NOT FOUND.
INPUT
```

OPEN Command referencing a nonexisting file

4.2.4 BLOCK ON/OFF Command

Form:

BLOCK␣{ON / OFF}⏎

Description:
This command (normally OFF) is used to establish a mode of operation in which the Editor sets up a buffer in core to retain a defined subset of the input file for edit operations. The size of the block "buffer" may be set by the user during the Edit "SIZE" command or a default size of 55_{10} lines may be accepted.

Once established, the block buffer is loaded by a "READ" command; all normal editing operations may then be performed on the contents of the buffer. On completion of editorial operations, the contents of the buffer are output onto the output file by a WRITE command. A RENEW command which writes the contents of the buffer onto the output file and then loads the next subset of the input file into the buffer may also be used.

The Block Mode may be turned ON or OFF at any time during the edit operations. Once the size of the buffer is established, it will remain in effect until the Editor is restarted or another SIZE command is issued.

NOTE

If either the input or the scratch file device is a non-directoried device, the Block Mode is automatically turned ON by Edit. If both input and scratch devices are directoried devices, Block Mode is set OFF when Edit is loaded.

The command

BLOCK⏎

(without arguments) is equivalent to

BLOCK␣␣ON⏎

Example:
1) Assuming that the example file EXAMP SRC is on the input device and is opened, the following commands set up a Block Mode with a 4-line buffer and loader and prints out the buffer:

```
 ↑P
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>BLOCK ON                          set up BLOCK Mode
>S 4                               set SIZE of buffer to 4 lines
>READ                              load buffer from input file (EXAMP SRC)
>P 5                               PRINT contents of buffer
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
        CCC
>
```

CONTENTS OF BLOCK MODE BUFFER

2) When the Block Mode is OFF, the entire input file is loaded, line-by-line, into the Editor work area; this is illustrated by the following:

```
EDIT
>WRITE                             empty buffer
>BLOCK OFF                         turn BLOCK mode OFF
>T                                 go to the top of the file in core
>P 20                              PRINT up to 20 lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
        CCC
TAG2    DDD     /LINE 4
        EEE
        FFF
TAG3    GGG     /LINE 7
        HHH
        III
        .END
END OF FILE REACHED BY:            message printed to indicate that the
P 20                                 end of the file was reached before
>                                    20 lines were printed.
```

printout of file loaded in core for Edit operations

4.2.5 SIZE [S] Command

Form:

SIZE␣N) where N is a decimal number which specifies the number of lines which the Block Mode buffer is to contain.

NOTE:

The name SIZE may be abbreviated as "S" (e.g., S␣N))

Description:

This command is effective only during Block ON Mode operations. It enables the user to specify, in number of lines, the size of the current Block Mode buffer. A size greater than 1 must be specified or an error will occur.

If a SIZE command is not used, the Editor program assumes a default size of 55_{10} lines. The default size is restored each time the Editor is restarted.

Whenever a BLOCK ON command is given, it should be followed by a SIZE command unless the default size is acceptable.

Example:

The command

a) S␣5 establishes a 5-line buffer
b) S␣100 establishes a 100-line buffer

4.2.6 VERIFY [V] ON/OFF Command

Form:

VERIFY␣{ON / OFF}↵

NOTE:

The name VERIFY may be abbreviated as "V" (e.g., V␣ON↵)

Description

The VERIFY command (normally ON) enables the user to determine whether or not a response is printed for certain Edit commands to verify that they were carried out. Some of the responses printed are:

a) the line brought into the work area as the result of a FIND or LOCATE command is printed
b) the last line of a file or block buffer brought in by the BOTTOM command is printed
c) the new line resulting from a CHANGE request is printed.

With the VERIFY mode OFF, only error messages are printed.

The command:

VERIFY↵ (or V ↵)

is equivalent to

VERIFY␣␣ON (or V␣␣ON ↵)

Example:

The following example illustrates the function of the VERIFY mode:

```
          .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1      AAA     /LINE 1
          BBB                          } current contents of block buffer
          CCC
>T                              go to top of buffer
>VERIFY ON                      turn Verify Mode ON
>F TAG                          find line in buffer containing string "TAG"
TAG1      AAA     /LINE 1       total line is printed to Verify operation
>T                              go to Top of buffer
>VERIFY OFF                     turn Verify Mode OFF
>F TAG                          repeat FIND command
>                               a go-ahead symbol is printed to indicate
                                operation was completed, the line is not
                                printed.
```

4.2.7 BRIEF ON/OFF Command

Form:

BRIEF␣␣{ON / OFF}↵

Description:

This command (normally OFF) enables the user to determine if the printed response to certain Edit commands is printed in a full (BRIEF OFF) or an abbreviated (BRIEF ON) form. For example, response made to the FIND, LOCATE, and BOTTOM commands will contain only items in the tag, operation code, and address fields, if the BRIEF mode is ON.

The VERIFY mode must be ON for the BRIEF mode command to have any effect on printed response.

The command given without an ON/OFF argument (e.g., BRIEF↵) is equivalent to a BRIEF␣␣ON↵ command.

Example:

The following example illustrates the manner in which the BRIEF mode affects the response to the command F␣␣TAG↵:

```
 VERIFY ON
>BRIEF OFF                          turn Brief Mode OFF
>T                                  go to Top of buffer
>F TAG                              find line containing the string "TAG"
TAG1     AAA      /LINE 1          total line is printed to Verify operation
>T                                  go to Top of buffer
>BRIEF ON                          turn Brief Mode on
>F TAG                              repeat Find command
TAG1     AAA                        an abbreviated line is printed to
>                                     Verify operation
```

4.2.8 OUTPUT ON/OFF Command

Form:

OUTPUT␣{ON / OFF})

Description:

The OUTPUT command is always ON when the Editor is loaded into core, and during normal Edit operations. When set to OFF, the OUTPUT command enables all Edit commands to be used; however, no data will be output to the output or scratch devices on .DAT-15. Also a "NOTHING IN FILE" message is output on the console printer if a WRITE or CLOSE command is issued.

This command may be issued at any time during file edit operations. Its use enables the operator to go through a file and output only selected portions of the file to what will be the final or new file when a CLOSE command is issued.

Example:

The following example illustrates the use of the OUTPUT command; as shown, this command is used to delete a segment of the file being edited.

```
 ↑P
 EDITOR V18A
>OPEN EXAMP SRC                    open demonstration program
EDIT
>P 12                              Print 12 lines of current file
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
        CCC
TAG2    DDD     /LINE 4
        EEE                           current (original) contents
        FFF                           of file being edited.
TAG3    GGG     /LINE 7
        HHH
        III
        .END
>T                                 go to Top of file
>F TAG2                            Find line starting with "TAG2"
TAG2    DDD     /LINE 4            printout of selected line
>OUTPUT OFF                        OUTPUT "OFF" command
>F TAG3                            Find line starting with "TAG3"
TAG3    GGG     /LINE 7            printout of selected line
>OUTPUT ON                         OUTPUT "ON" command
>B                                 go to Bottom of file
        .END                       printout of last line in file
>CLOSE                             Close file and terminate Edit operations

 EDITOR V18A
>OPEN EXAMP SRC                    reopen demonstrator program
EDIT
>P 12                              Print 12 lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
        CCC                           contents of the demonstrator
TAG3    GGG     /LINE 7               program as modified by the
        HHH                           OUTPUT ON/OFF command in the
        III                           previous Edit operation.
        .END
END OF FILE REACHED BY:
P 12
>
```

4.2.9 KEEP [K] Command

Form:

KEEP␣savnam␣ext) where "savnam" = the name of the file to be saved. If an extension is not given, "SRC" is assumed.

NOTE

KEEP may be abbreviated as "K" (e.g., K␣savnam␣ext)).

Description:

This command causes the file on .DAT-14 when it is issued, to be preserved for back-up purposes. As many KEEP commands may be issued as desired; however, each "savnam" must be different from any other file name on the device at .DAT-14. Normally, the KEEP command may be issued only while the input file is at .DAT-14 (i.e., the numbers of TOPs issued must be even, 0, 2, 4,...). However, if the same device and UIC are assigned to both the I/O and scratch functions (such as operating with a file in a DOS UFD) the number of TOPs issued with the KEEP command need not be even. The actual renaming of the input file, however, is not done until the next TOP, CLOSE, EXIT, or ICLOSE command is issued.

Example:

The following example illustrates the use of the KEEP command in a DOS system to maintain both the original and edited files:

```
   02-OCT-71
DIRECTORY LISTING  (JAN)
  2437 FREE BLKS
     1 USER FILES
     1 USER BLKS
EXAMP  SRC       1  02-OCT-71
```

listing of UFD on input .DAT slot

note single file entry

```
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>KEEP EXAMP                          keep file "EXAMP SRC" intact on input dev.
>F TAG                               Find file line beginning "USING"
TAG1    AAA     /LINE 1              printout verifying Find operation
>T                                   go to the Top of the file
FILE EXAMP SRC IS PRESENT ON INPUT DEVICE.  request for new name for output
PLEASE TYPE IN ANOTHER NAME.                file initiated by T (TOP) com'd
>EXAMP2                              new name
>CLOSE                              Close file and restart Editor

 EDITOR V18A
>
```

```
   02-OCT-71
DIRECTORY LISTING  (JAN)          } listing of UFD on
  2436 FREE BLKS                  } input .DAT slot after
     2 USER FILES                 } Edit operations
     2 USER BLKS
EXAMP2 SRC       1  02-OCT-71 --  - original file
EXAMP  SRC       1  02-OCT-71 --  - new file
```

4.3 INPUT/OUTPUT COMMANDS

The Edit Mode commands which are used to input and output data during editing operations are described in paragraphs 4.3.1 through 4.3.4.

4.3.1 READ Command

Form:

READ)

Description:
The READ command is permitted in the BLOCK ON mode of operation only. This command fills the Block buffer with sequential lines from the input file. Once filled, the buffer must be emptied before another READ command can be issued (see WRITE and RENEW). The number of input file lines read into the current block buffer is determined either by the value specified in a previously entered SIZE (S) command or, if S is not given, the default value (i.e. 55_{10}) is used.

On the completion of a READ operation, the current line pointer is positioned one line above the first line of the buffer.

Example:
The following illustrates the manner in which the READ command is used to load a four-line Block Buffer.

```
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>BLOCK ON                    turn BLOCK mode ON
>S 4                         setup a four-line block buffer
>READ                        fill buffer from input file
>P 5                         PRINT five lines
        .TITLE  EXAMP        /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB                                  contents of
        CCC                                  block buffer

>WRITE                       WRITE buffer contents on output/scratch file
>READ                        fill buffer with next sequential set of lines
>P 5                         PRINT five lines
TAG2    DDD     /LINE 4
        EEE                                  contents of
        FFF                                  block buffer
TAG3    GGG     /LINE 7
>
```

4.3.2 WRITE Command

Form:

WRITE)

Description:
This command is permitted in only BLOCK ON modes of operation. It causes the contents of the current block buffer to be written onto the output file and then clears the buffer. The WRITE Command may be issued at any time during a Block mode Edit operation.

4.3.3 RENEW Command

Form:

RENEW)

Description:
This command is permitted only during BLOCK ON modes of operation. The RENEW command causes a buffer write-clear-fill sequence of

operations to be performed. This command replaces a WRITE, READ command sequence.

On the completion of a RENEW operation, the current line pointer is positioned one line above the first line of the buffer.

Exammple:

```
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>BLOCK ON                          turn BLOCK mode ON
>S 4                               set up a 4-line buffer
>READ                              fill buffer from input file
>P 5                               PRINT 5 lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1                                  } contents of buffer
        BBB                                              }
        CCC                                              }
>RENEW                             write, clear and fill buffer
>P 5                               PRINT 5 lines
TAG2    DDD     /LINE 4                                  }
        EEE                                              } new contents of buffer
        FFF                                              }
TAG3    GGG     /LINE 7                                  }
>
```

4.3.4 GET [G] Command

Form:

GET␣n) where n is a decimal number.

NOTE:

GET may be abbreviated as "G" (e.g., G␣n)).

Description:

This command causes n lines from the subsidiary input device to be added to the output file. New lines are added below the current line. When command processing is complete, the n^{th} line read is left in the work area as the current line. If n is omitted, it is assumed to be 1.

If an end-of-medium condition is encountered on the subsidiary input device before n lines are read, the error message

END OF MEDIUM REACHED BY:

GET n

is printed. The pointer remains at the last line read.

Example:
The following printout illustrates the manner in which data from a subsidiary input device is inserted into a file using the GET command.

```
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>L TAG2                          LOCATE line containing string "TAG2"
TAG2    DDD     /LINE 4          verification printout
>GET 3                           GET 3 lines from subsidiary dev; insert in file
>T                               go to TOP of file
>P 20                            PRINT 20 lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
        CCC
TAG2    DDD     /LINE 4                                  inserted lines
THIS MESSAGE IS BEING ADDED TO THE OUTPUT FILE          }
FROM THE SUBSIDARY INPUT DEVICE (PR IN THIS CASE)       }
USING THE GET COMMAND                                    contents of file
        EEE
        FFF
TAG3    GGG     /LINE 7
        HHH
        III
        .END
END OF FILE REACHED BY:
P 20
>
```

4.4 LINE POINTER CONTROL (LOCATIVE) COMMANDS

During editing operations, the Editor program maintains a pointer which identifies the current line (i.e., the line which any subsequent editing operations will affect or will be referenced to).

Commands are provided which permit the user to control the positioning of the "current line" line pointer and to cause the pointer to be set to a line identified only by a text string contained by the line. The commands provided enable the user to: 1) set the line pointer to either the Top or Bottom of a file or buffer, 2) move the line pointer a specified number of lines away from its current position, and 3) make a line, identified only by a text string, the current line.

4.4.1 TOP [T] Command

Form:

TOP)

NOTE

This command may be abbreviated as "T" (e.g., T).

Description:

The T command causes the line pointer to be moved to one line position above the first line of the file or buffer being edited. The pointer is positioned in this manner to enable the user to insert, if desired, data before the first line of the file or buffer. The TOP command may be entered at any time during the Edit operations.

Example:

The following example illustrates the use of the TOP command and the fact that it is positioned above the current file or buffer.

```
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>L TAG2                          LOCATE line in file containing string "TAG2"
TAG2    DDD      /LINE 4         verification printout
>TOP                             go to TOP of file
>P                               PRINT current line (nothing is printed since
>N       go to NEXT line           pointer is above first line of the file)
>P                               PRINT current line
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR   first line in file
>
```

4.4.2 NEXT [N] Command

Form:

NEXT␣n) where n is a decimal number.

NOTE

This command may be abbreviated as "N" (e.g., N␣n))

Description:

This command enables the user to move the line pointer a specified number (i.e., n) of lines towards the bottom (end) of a file or

buffer. The NEXT operation begins with the line current when the command was issued and ends with the line pointer at the position "current line + n". During the line-by-line (File) mode, lines skipped over are added to the output file.

The N command may be issued at any time during Edit operations. If a value is not given for n, it is assumed to be 1.

If an N command results in the pointer being moved past the last line of the file or buffer, the error message:

END OF {FILE / BUFFER} REACHED BY:

NEXT n

is printed at the console.

Example:
The following illustrates the use of the NEXT command:

```
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>P 2                                  PRINT two lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR  first line of file
>NEXT 3                               move pointer three lines
>P                                    PRINT current line
        CCC                           current line
>N 2                                  move pointer two lines
>P                                    PRINT current line
        EEE                           current line
>N                                    move pointer one line
>P                                    PRINT current line
        FFF                           current line
>N 12                                 move pointer twelve lines
END OF FILE REACHED BY:               error message
N 12
>
```

4.4.3 BOTTOM [B] Command

Form:

BOTTOM␣)

NOTE:

This command may be abbreviated as "B" (e.g., B␣)).

Description:

The B command causes the line pointer to be moved to the final line of the file or buffer being edited. In a file mode of operation, any lines skipped over are added to the output file.

Example:

The following verifies that the B command will move the pointer to the last line of a buffer:

```
 ↑P
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>BLOCK ON                          turn BLOCK mode ON
>S 5                               setup a five-line buffer
>READ                              load buffer from input file
>P 6                               PRINT six lines (i.e.buffer contents)
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB                                              } current contents
        CCC                                                of buffer
TAG2    DDD     /LINE 4
>T                                 move pointer to TOP of buffer
>BOTTOM                            move pointer to last line of buffer
TAG2    DDD     /LINE 4            line printed to verify operation
>
```

4.4.4 FIND [F] Command

Form:

FIND␣string␣)

NOTE:

This command may be abbreviated as "F" (e.g., F␣string␣)).

Description:

The F command causes a sequential line-by-line search to be made of the file or buffer being edited for the first occurrence of a line

which begins with the specified "string". The search starts below the current line; if the specified line is found, it becomes the current line and is printed at the console. If a line starting with the given "string" is not found, the line pointer is positioned immediately below the last line of the file (or buffer) and the message:

END OF { FILE / BUFFER } REACHED BY:
FIND (or F) "string"

is printed at the console.

In entering an identifying "string", the user must ensure that it matches exactly the beginning of the line to be found. The command "string" must include all tabs and/or spaces preceding or within the first portion of the desired line.

Example:
The following illustrates the use of the "F" command.

```
 EDITOR V18A
>OPEN EXAMP SRC                 open demonstration program
EDIT
>FIND TAG3                      make line starting "TAG3" current
TAG3    GGG      /LINE 7        verification printout
>T                              go to TOP of file
>FIND .END                      make line starting ".END" current
END OF FILE REACHED BY:         error message, the line was not found
FIND .END                         (the user omitted the tab preceding .END)
>
```

4.4.5 LOCATE [L] Command

Form:

LOCATE␣string↵

NOTE

This command may be abbreviated as "L" (e.g., L␣string↵).

Description:
This command causes a sequential line-by-line search to be made of the file or buffer being edited for the next occurrence of a line containing the specified "string". The identifying "string" may be numeric, alphabetic, or alphanumeric and need not be a complete word. The "string" given, however, should be unique, within the file or buffer, to the desired line.

The initiated search starts at the current line; if the specified line is found, it becomes the current line and is printed at the console. If the identified line is not found, the line pointer is moved immediately below the last line of the file or buffer and the message:

END OF {FILE / BUFFER} REACHED BY:

LOCATE (or L) "string"

is printed at the console.

Example:
The following illustrates the use of the "L" command:

```
 EDITOR V18A
>OPEN EXAMP SRC              open demonstration program
EDIT
>LOCATE TAG3                 make line containing "TAG3" current
TAG3    GGG      /LINE 7     verification printout
>T                           go to TOP of file
>L G3                        make line containing "G3" current
TAG3    GGG      /LINE 7     verification printout
>T                           go to TOP of file
>L NE 7                      LOCATE line containing "NE 7"
TAG3    GGG      /LINE 7     verification printout
>T                           go to TOP of file
>L ZZZ                       make line containing "ZZZ" current
END OF FILE REACHED BY:      error message, no line containing "ZZZ"
L ZZZ                          was found
>
```

4.5 TEXT MODIFICATION AND MANIPULATIVE COMMANDS

The Edit Mode commands which enable the user to view, change data, and modify the structure of the current file or buffer are described in paragraphs 4.5.1 through 4.5.11. Explicit examples of the use of this type of Edit command are given wherever possible.

4.5.1 PRINT [P] Command

Form:

PRINT␣n) where n is a decimal number.

NOTE

This command may be abbreviated as "P" (e.g., P␣n)).

Description:
This command permits the user to print, at the console, one line or a group of "n" lines of the current file or buffer. Print operations start at the current line; the last line printed becomes the current line. If no value is given for "n", it is assumed to be 1; the command "P)" causes the current line to be printed; however, it will also remain the "current" line after being printed.

After a "T" TOP command, the user must specify n+1 lines to get the desired printout, since the line pointer will be one line above the first line of the file or buffer.

If the value of n is too large, it will cause the current file/buffer line pointer to be moved past the last line of the file/buffer and the message

END OF { FILE / BUFFER } REACHED BY:

P␣n

is printed at the console.

Example:
The following annotated listing illustrates the use of the "P" command.

```
 EDITOR V18A
>OPEN EXAMP SRC              OPEN demonstration program
EDIT
>PRINT 5                     PRINT 5 lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR  } lines printed.
TAG1    AAA     /LINE 1                             } note only four
        BBB                                         } lines printed
        CCC                                         } because line
>P 2                         PRINT 2 lines          } pointer was above
        CCC                                         } first file line
TAG2    DDD     /LINE 4     response
>P                           PRINT current line
TAG2    DDD     /LINE 4     response
>P 10                        PRINT ten lines
TAG2    DDD     /LINE 4
        EEE
        FFF                      all remaining file lines printed
TAG3    GGG     /LINE 7          in response to the P 10 command
        HHH
        III
        .END
END OF FILE REACHED BY:      } error message; the end of file was reached
P 10                         } before ten lines were printed
>
```

4.5.2 DELETE [D] Command

Form:

DELETE␣n⏎ where n is a decimal number.

NOTE

This command may be abbreviated as "D (e.g., D␣n⏎).

Description:

The D command enables the user to delete one line, or a group of "n" lines from the current file or buffer. If a value for n is not given, the value 1 is assumed. Whenever a line is deleted, the line pointer is moved to the next sequential line. If the value of n is given too large, the pointer is moved below the last line in the file or buffer, an end-of-file/buffer message is printed at the console.

Example:

The following example illustrates the use of the "D" command.

```
 EDITOR V18A
>OPEN EXAMP SRC              OPEN demonstrator program
EDIT
>BLOCK ON                    turn BLOCK mode ON
>S 6                         establish a six line buffer
>READ                        load buffer
>P 7                         PRINT contents of buffer
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
        CCC
TAG2    DDD     /LINE 4                          } contents of
        EEE                                      } block buffer
>T                           go to TOP of buffer
>L BBB                       LOCATE line containing "BBB"
        BBB                  verification printout
>DELETE                      DELETE current line
>T                           go to TOP of buffer
>P 7                         PRINT contents of buffer
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1                          }
        CCC                                      } contents of
TAG2    DDD     /LINE 4                          } block buffer;
        EEE                                      } note line con-
END OF BUFFER REACHED BY:                        } taining BBB
P 7                                              } is gone
```

```
>T                                  go to TOP of buffer
>L CCC                              LOCATE line containing "CCC"
      CCC                           verification printout
>D 3                                DELETE 3 lines including current line
END OF BUFFER REACHED BY:
D 3
>T                                  go to TOP of buffer
>P 6                                PRINT contents of buffer
      .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1  AAA     /LINE 1                                   } contents of
END OF BUFFER REACHED BY:                               } buffer after
P 6                                                     } DELETE operation
>
```

4.5.3 RETYPE [R] Command

Form:

RETYPE␣line↵

NOTE

This command may be abbreviated as "R: (e.g., R␣line↵).

Description:

This command causes the current line to be replaced, entirely, by the "line" given in the command.

Example:

The following illustrates the use of the "R" command:

```
 EDITOR V18A
>OPEN EXAMP SRC                     OPEN demonstration program
EDIT
>P 6                                PRINT six lines
      .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1  AAA     /LINE 1                                   } first six lines
      BBB                                               } of file
      CCC
TAG2  DDD     /LINE 4
>T                                  go to TOP of file
>L BBB                              LOCATE line containing "BBB"
      BBB                           verification printout
>RETYPE THIS MESSAGE REPLACES LINE BBB     RETYPE command
>T                                  TOP command
>P 6                                PRINT six lines
      .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1  AAA     /LINE 1                                   } first five lines
THIS MESSAGE REPLACES LINE BBB                          } of file; note
      CCC                                               } change in line
TAG2  DDD     /LINE 4                                   } "BBB"
>
```

4.5.4 INSERT [I] Command

Form:

INSERT↵

NOTE

This command may be abbreviated as "I" (i.e., I↵).

Description:

This command enables the user to insert via the keyboard, one or more lines between selected lines of the current file or buffer. When the I command is issued and the operating mode is changed from EDIT to INPUT, the user may key in any desired data. The INPUT operation is terminated by the entry of a blank line (i.e., a line containing only a ↵). Once the EDIT mode is restored, the current line pointer will be positioned at the last line entered when in input mode.

Example:

The following illustrates the use of the I command.

```
 EDITOR V18A
>OPEN EXAMP SRC                    OPEN demonstrator program
EDIT
>P 6                               PRINT six lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1                                  } first five lines
        BBB                                              } of current file
        CCC                                              }
TAG2    DDD     /LINE 4                                  }
>T                                 TOP
>L BBB                             LOCATE line containing "BBB"
        BBB
>INSERT                            INSERT command
INPUT                              INPUT mode identifier
1111    ADDED LINE                 }
2222    ADDED LINE                 } inserted material
3333    ADDED LINE                 }

EDIT                               EDIT mode identifier
>T                                 TOP
>P 10                              PRINT ten lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
1111    ADDED LINE                                       } contents of file;
2222    ADDED LINE                                       } note inserted
3333    ADDED LINE                                       } material
        CCC
TAG2    DDD     /LINE 4
        EEE
>
```

4.5.5 INSERT [I] Line Command

Form:

INSERT␣␣line)

NOTE

This command may be abbreviated as "I" (e.g., I␣␣line)).

Description:

This command enables the user to insert single "lines" between lines of a current file or buffer without changing the operating mode. When an I␣␣line command is issued, the current line is added to the output file and the command "line" data becomes the current line. Note that the insertion is made below the line current when the command is issued. The operation remains in the Edit Mode throughout the execution of this command.

Example:

The following illustrates the use of the I␣␣line command.

```
 EDITOR V18A
>OPEN EXAMP SRC              OPEN demonstrator program
EDIT
>P 6                         PRINT six lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
        CCC
TAG2    DDD     /LINE 4
                             TOP
>T                           LOCATE line "BBB"
>L BBB
        BBB
>INSERT 1111 INSERTED LINE   INSERT command
                             TOP
>T
>P 10
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
1111 INSERTED LINE
        CCC
TAG2    DDD     /LINE 4
        EEE
        FFF
TAG3    GGG     /LINE 7
>
```

first five lines of current file

contents of current file; note inserted line immediately after line "BBB"

4.5.6 CHANGE [C] Command

Form:

CHANGE␣/string 1/string 2/↵

NOTE

This command may be abbreviated as "C" (e.g., C␣/S1/S2/↵); also, the use of the final separator (/) is optional. Delimiters other than / may be used; any character (including blanks) which does not appear in string1 or string2 may be used as delimiter.

Description:

This command enables the user to replace selected portions of a current line. Part "/string1/" of the command identifies the portion of the current line to be replaced; part "/string 2/" specifies the new data to be entered. Both /string 1/ and /string 2/ may contain any number of characters, including Ø.

Null (blank) spaces to be replaced or entered are indicated by two consecutive separators (i.e., /'s) such as //.

If the number of characters in the current line, including the terminator, exceeds 90_{10} as the result of a CHANGE command, the message:

TRUNCATED

is printed at the console and the excessive right-hand characters of the current line are discarded.

When the VERIFY Mode is ON, the changed current line is automatically printed at the console on completion of the CHANGE operation.

Example:

The following illustrates the use of the C command.

```
 EDITOR V18A
>OPEN EXAMP SRC              OPEN demonstration program
EDIT
>P 4                         PRINT four lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR  } first three
TAG1    AAA     /LINE 1                             } lines of
        BBB                                         } current file
>T                           TOP
>L TAG                       LOCATE line containing "TAG"
TAG1    AAA     /LINE 1
>CHANGE /TAG/ADD/            CHANGE "TAG" to "ADD" in current line
ADD1    AAA     /LINE 1      verification printout
>T
>L .T                        LOCATE line containing ".T"
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
>C /TOR/TION/                                       CHANGE command
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATION changed line
>T
>L BBB                       LOCATE line "BBB"
        BBB
>C //TAGB/                   add "TAGB" to line using CHANGE command
TAGB    BBB
>T
>P 4
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATION } first three
ADD1    AAA     /LINE 1                             } lines of modified
TAGB    BBB                                         } file
>T                           TOP
>L TAGB                      LOCATE line containing "TAGB"
TAGB    BBB
>C /TAGB//                   use null string to delete "TAGB"
        BBB
>
```

4.5.7 OVERLAY [O] Command

Form:

OVERLAY␣n) where n is a decimal number.

NOTE

This command may be abbreviated as "O" (e.g., O␣n).

Description:

This command enables the user to delete one or more lines and to insert lines input from the keyboard in place of the deleted material. The number of lines to be deleted in an "O" operation is specified by the value assigned to n. If n is omitted from the command, 1 is assumed. When an "O" command is issued, the indicated lines are deleted and the operating mode is changed from EDIT to INPUT.

The user may then enter any number of desired lines to replace the deleted lines. The INPUT mode is terminated and the EDIT mode restored by the entry of a blank line (i.e., a line containing only a ↵ or ALT MODE entry.

Example:
The following illustrates how the O command is used to delete one line and replace it with two lines input from the console keyboard.

```
 EDITOR V18A
>OPEN EXAMP SRC                    OPEN demonstrator program
EDIT
>P 5                               PRINT five lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1                                  } first four lines
        BBB                                              } of current file
        CCC                                              }
>T
>L CCC                             LOCATE line "CCC"
        CCC
>OVERLAY                           OVERLAY line "CCC"
INPUT                              INPUT mode identifier
THIS INPUT LINE IS TO REPLACE (OVERLAY) LINE CCC         } keyed in
THIS LINE IS TO BE INSERTED BETWEEN NEW LINE AND TAG2    } lines

EDIT                               restore EDIT mode
>T
>P 8
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR       } first six lines
TAG1    AAA     /LINE 1                                  } of current file
        BBB                                              } showing the
THIS INPUT LINE IS TO REPLACE (OVERLAY) LINE CCC         } input overlays
THIS LINE IS TO BE INSERTED BETWEEN NEW LINE AND TAG2    }
TAG2    DDD     /LINE 4                                  }
        EEE
>
```

4.5.8 APPEND [A] Command

Form:

APPEND␣string↵

NNOTE

This command may be abbreviated as "A" (e.g., A␣string↵).

Description:
The A command enables the user to add (append) data to the current line. The appended data will be placed after the last character of the

original line and may contain spaces and tabs as well as alphanumeric characters.

Example:
The following illustrates the use of the A command:

```
 EDITOR V18A
>OPEN EXAMP.SRC                    OPEN demonstrator program
EDIT
>P 6
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR   } first five
TAG1    AAA     /LINE 1                              } lines of
        BBB                                          } current file
        CCC                                          }
TAG2    DDD     /LINE 4                              }
>T
>L BBB                             LOCATE line "BBB"
        BBB
>APPEND         /LINE 2            add " tab /LINE 2" to current line
>N
>P
        CCC
>A      /LINE 3                    add " tab /LINE 3" to current line
>T
>P 6
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR   }
TAG1    AAA     /LINE 1                              }
        BBB     /LINE 2                              } printout of first
        CCC     /LINE 3                              } five file lines
TAG2    DDD     /LINE 4                              } showing added
>                                                    } material
```

4.5.9 MOVE Command (DOS SYSTEMS ONLY)

Form:

MOVE␣n␣string) where n is a decimal number

Description:
The MOVE command enables the user to move the current line or n lines including the current line from their position in the file or buffer to a position immediately above (preceding) the file/buffer line identified by the command "string". Any desired decimal value may be assigned to n; it may not be omitted from the command.

If the value of n is too large, an END of File Reached message is given but nothing is moved from its original position.

The command "string" should be the first characters of the line; where spaces and tabs are used as field delimiters, they must be entered as part of the identifying text "string".

Example:
The following example illustrates the use of the MOVE command.

```
 EDITOR V18A
>OPEN EXAMP SRC              OPEN demonstrator program
EDIT
>P 6
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR  } printout of
TAG1    AAA     /LINE 1                             } first five
        BBB     /LINE 2                             } file lines
        CCC     /LINE 3
TAG2    DDD     /LINE 4
>T
>L TAG                       LOCATE line containing "TAG"
TAG1    AAA     /LINE 1      current (LOCATEd) line
>MOVE 1 TAG2                 MOVE current line to position above line TAG2
>T
>P 6
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR  } printout of
        BBB     /LINE 2                             } first five file
        CCC     /LINE 3                             } lines; note
TAG1    AAA     /LINE 1                             } moved line
TAG2    DDD     /LINE 4
>
```

During a BLOCK ON state:

a) If the number of lines specified in the MOVE command exceeds or equals the size of the current block buffer, the Editor outputs the message:

SIZE TOO SMALL

on the console printer and waits for the next user entry.

Example:

```
 ↑P
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>BLOCK ON
>S 5
>READ
>L TAG1
TAG1    AAA     /LINE 1
>MOVE 10 TAG2
SIZE TO SMALL
>P
TAG1    AAA     /LINE 1          note, current line is unchanged
>
```

b) If the MOVE command specifies more lines to be moved than are in the block buffer, EDIT outputs the error message:

```
NOT ENOUGH LINES
>
```

on the console printer and waits for the next user entry.

NOTE

Neither of the above error conditions changes the position of the current line pointer or affects the contents of the block buffer.

During a BLOCK MODE OFF state:

If the MOVE command specifies more lines to be moved than exist in the current file, EDIT outputs the error message:

```
NOT ENOUGH LINES
END OF FILE REACHED BY:
(MOVE command entered)
```

on the console printer and waits for the next user entry.

Example:

```
 ↑P
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>L BBB
        BBB     /LINE 2
>MOVE 15 TAG3
NOT ENOUGH LINES
END OF FILE REACHED BY:
MOVE 15 TAG3
>
```

If there is not enough buffer space to store the lines being moved, the Editor outputs the message:

NOT ENOUGH BUFFER SPACE

at the console printer and waits for the next user entry. Neither of the above conditions affects the lines specified in the MOVE command; however, in an "END OF FILE" error condition, the current line pointer is moved past the last line in the file and there will be no "current line". The user must then enter a "TOP" command and specify another line with a FIND or LOCATE command.

Whenever a "NOT ENOUGH BUFFER SPACE" error occurs, the current line pointer is moved to the line which caused the overflow condition. If the line containing the text string specified in the MOVE command is not found, EDIT outputs the error message:

LINES MOVED TO END
END OF {BUFFER / FILE} REACHED BY:
(MOVE COMMAND)

and waits for the next user entry.

When an error of this type occurs, the lines to be moved are deleted from their original positions in the file and are placed at the end of the file. The lines are not lost but it is difficult to recover from this condition, care should be taken in entering the MOVE command (text string) to prevent this error from occurring...

Example:
In the following example, line "TAG1" was erroneously moved to below the last line of the file when it was to be placed after the ".TITLE" line.

The user recovered from this error by utilizing the MOVE command to move the 9 lines between ".TITLE" and "TAG1" lines to a false position identified by string "ZZZ". The command results in the specified 9 lines being moved below the last line of the file (i.e., "TAG1") thus correctly positioning all lines of the file.

```
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>L TAG1
TAG1    AAA     /LINE 1
>MOVE 1 BBB                    bad command; tab before "BBB" not included
LINES MOVED TO END
END OF FILE REACHED BY:        resulting error message
MOVE 1 BBB
>T
>P 12
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
        BBB
        CCC
TAG2    DDD     /LINE 4
        EEE
        FFF
TAG3    GGG     /LINE 7
        HHH
        III
        .END
TAG1    AAA     /LINE 1        resulting position of line "TAG1"
>T
>L BBB
        BBB
>MOVE 9 ZZZ                    MOVE lines "BBB" through ".END" to
LINES MOVED TO END             end of file, i.e., below line "TAG1"
END OF FILE REACHED BY:
MOVE 9 ZZZ
>T
>P 12
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
        CCC
TAG2    DDD     /LINE 4
        EEE
        FFF
TAG3    GGG     /LINE 7
        HHH
        III
        .END
>
```

printout of resulting file

NOTE

The above recovery procedure is acceptable for short files; however, when editing long files it is more realistic to enter a CTRL P (Editor restart) and to reopen the file. A CTRL P command entered during the processing of the MOVE command is ignored. After the MOVE has been processed then a new CTRL P command will be honored.

4.5.10 CONVERT Command (DOS SYSTEMS ONLY)

Form:

CONVERT␣/string 1/string 2/↵

NOTE

The use of the last separator (/) is optional. Any character not in string1 or string2 may be used as a delimiter.

Description:
The CONVERT command enables the user to change a specific item or data set which is repeated throughout a file using only one command.

In the execution of this command, EDIT scans the entire buffer or file (BLOCK mode ON/OFF) and each time that it finds a line string which matches "string 1" it replaces that string (converts) with that of "string 2" of the command.

If a "string 2" is not given in the CONVERT command, all line strings matching the given "string 1" are deleted from the buffer (or file). On completion of a CONVERT command, the "current line pointer" points to the first line below the last line in the file or buffer (i.e. End Of File/Buffer condition).

If, during the convert process, the current line overflows (i.e., exceeds 90 characters including the terminator), it is truncated and the following message is output at the console printer:

```
(truncated line)
TRUNCATED
```

After outputting the above the Editor continues with the convert operation.

A "string 1" must be given in the CONVERT command, otherwise the message:

```
NO STRING
```

is output at the console printer and the Editor waits for the next user command.

Example:
The following illustrates the use of the CONVERT command.

```
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>P 12
          .TITLE   EXAMP    /EDIT COMMAND DEMONSTRATOR
TAG1      AAA      /LINE 1
          BBB
          CCC
TAG2      DDD      /LINE 4
          EEE
          FFF
TAG3      GGG      /LINE 7
          HHH
          III
          .END
>T
>CONVERT /TAG/ADD/
>T
>P 12
          .TITLE   EXAMP    /EDIT COMMAND DEMONSTRATOR
ADD1      AAA      /LINE 1
          BBB
          CCC
ADD2      DDD      /LINE 4
          EEE
          FFF
ADD3      GGG      /LINE 7
          HHH
          III
          .END
>
```

original file

converted file

IF "string 1" is never found, then the message

NOT FOUND

is output on the console printer and the Editor waits for the next user command.

4.5.11 LC (Line Convert) Command (DOS SYSTEM ONLY)

Form:

LC␣n␣/string 1/string 2/)

NOTE

The use of the last separator (/) in the command is optional. Any character not in string 1 or string 2 may be used as a delimiter in this command.

Description:

This command performs the same type of operation as the CONVERT command (see 4.5.10); however, it enables the user to specify the number (n) of lines to be scanned during the convert operation.

The value assigned n indicates the number of lines to be scanned including the current line. The last line scanned is printed and is made the current line. The value n cannot be omitted from the command.

If the end of the file (or buffer) is reached by the LC command, the message:

```
END OF FILE
       BUFFER   REACHED BY

LC /STRING1     (Note command line starting with
                 second separator is not typed)
```

is typed on the console.

```
 EDITOR V18A
>OPEN EXAMP SRC
EDIT
>P 12
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1
        BBB
        CCC
TAG2    DDD     /LINE 4
        EEE
        FFF
TAG3    GGG     /LINE 7
        HHH
        III
        .END
>T
>LC 5 /TAG/ADD                  Line Convert command
ADD2    DDD     /LINE 4         current line after convert operation
>T
>P 12
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
ADD1    AAA     /LINE 1         converted line
        BBB
        CCC
ADD2    DDD     /LINE 4
        EEE
        FFF
TAG3    GGG     /LINE 7
        HHH
        III
        .END
>
```

original file

converted file

4.5.12 MODIFY [M] Command (DOS SYSTEMS ONLY)

Form:

MODIFY␣␣string) or ALT MODE

NOTE

This command may be abbreviated as "M" (e.g., M␣␣string) or ALT MODE).

Description:

This command causes the contents of the current line up to the last character of the specified "text string" to be printed at the console. The user may then add characters to the current line via the console keyboard. The input operation may be terminated with either:

a) an ALT MODE character which specifies that the remainder (unprinted portion) of the original line is to be added to the modified line;
b) a RETURN ( ) character which specified that the unprinted portion of the original line is to be deleted.

A line may contain a maximum of 90 characters (including terminator); when this limit is reached during MODIFY operations, the Editor performs a RETURN operation and outputs a go-ahead symbol (>). Care must be taken to ensure that the final line size does not exceed 90 characters since characters above this limit are dropped from the final line.

NOTE

The RUBOUT and CTRL U commands may be used for data keyed in during MODIFY operations; however, they will not affect the contents of the original line.

CTRL P should not be used during MODIFY operations; if the user wishes to go back to the original line, CTRL U should be entered to delete any new added data and the operation terminated by ALT MODE to restore the original line.

Example:

```
 EDITOR V18A
>OPEN EXAMP.SRC                    OPEN demonstrator program
EDIT
>P 12
          .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1      AAA     /LINE 1
          BBB
          CCC
TAG2      DDD     /LINE 4
          EEE
          FFF
TAG3      GGG     /LINE 7                                  original
          HHH                                               file
          III
          .END
>T
>L TAG2
TAG2      DDD     /LINE 4
>MODIFY DDD                        MODIFY command
TAG2      DDD     NEW DATA WITH RETURN TERMINATOR    modified line
>P                                                   new data
TAG2      DDD     NEW DATA WITH RETURN TERMINATOR    new form of line TAG2
>L TAG3
TAG3      GGG     /LINE 7
>MODIFY GGG
TAG3      GGG     NEW DATA WITH ALT MODE TERMINATOR        /LINE 7   modified
>P                                                   new data        line
TAG3      GGG     NEW DATA WITH ALT MODE TERMINATOR        /LINE 7   new form
>B
          .END
>CLOSE

 EDITOR V18A
>
```

4.6 EDIT CLOSE OPERATION COMMANDS

The EDIT Mode commands which are used to terminate EDIT operations and to write the edited program onto the final output device are described in paragraphs 4.6.1 through 4.6.4.

4.6.1 CLOSE Command

Form:

CLOSE␣filename␣extension)

Description:
If an input file is present, the current line and all lines in that file falling below the current line are appended to the output file, and the output file is closed. If no input file is present, the current line is added to the output file, and the output file is closed. No further editing is permitted.

If the extension is omitted, it is assumed to be SRC. If no file name is given, the name assigned in the OPEN command line is used

Neither file name nor ext need be given for nonfile-oriented output device.

4.6.2 ICLOSE Command

Form:

ICLOSE)

Description:
The ICLOSE command effects the closing of the current input file only. The output file remains open. A new input file can be referenced after the ICLOSE request by issuing an OPEN command. ICLOSE provides a facility for combining source files during an editing run.

```
 EDITOR V18A
>OPEN EXAMP SRC              OPEN demonstrator program
EDIT
>P 5                         PRINT five lines
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR
TAG1    AAA     /LINE 1                                  } first four
        BBB                                              } file lines
        CCC
>ICLOSE                      CLOSE Input file, retain current output file
>OPEN FILEA SRC              OPEN file named "FILEA SRC"
EDIT
>P 5                         PRINT five lines of file FILEA SRC
        CCC    line current in previously opened file when ICLOSE issued
        .TITLE ICLOSE EXAMPLE                            } first four lines
TAGA    1111                                             } of file FILEA SRC
TAGB    2222                                             }
TAGC    3333                                             }
>CLOSE

 EDITOR V18A
>OPEN FILEA SRC              OPEN modified file
EDIT
>P 10
        .TITLE  EXAMP   /EDIT COMMAND DEMONSTRATOR       }
TAG1    AAA     /LINE 1                                  }
        BBB                                              }
        CCC                                              } contents of file
        .TITLE ICLOSE EXAMPLE                            } modified using
TAGA    1111                                             } ICLOSE command
TAGB    2222                                             }
TAGC    3333                                             }
        .END                                             }
>
```

4.6.3 SCLOSE Command

Form:

SCLOSE ↵

Description:
This command permits the placement of an edited file onto the current output device without the .DAT-14 to .DAT-15 recopy process. It is particularly useful in closing long files which have only minor changes.

In employing the command SCLOSE, always use a file name different from that used with the OPEN command given for the file. Files closed in this manner are normally left on .DAT-15.

4.6.4 EXIT [E] Command (DOS SYSTEMS ONLY)

Form:

EXIT ↵

NOTE

This command may be abbreviated as "E" (e.g., E↵).

Description:
The EXIT command performs the same operations as the CLOSE command except that control is returned to the Monitor after the file is closed instead of to the EDITOR program.

Example:

```
 EDITOR V18A
>OPEN FILEA SRC
EDIT
>P
BBBB
>EXIT


DOS-15 V1A
$
```

APPENDIX A

SUMMARY OF EDITING COMMANDS

COMMAND	ABBREVIATION	OPERATION	PARAGRAPH REFERENCE
APPEND	A	Add "string at the rightmost end of the current line.	4.5.8
BLOCK { ON / OFF }	none	Set program to operate in block mode (ON) or in line-by-line mode (OFF).	4.2.4
BOTTOM	B	Bring last line of file to work area.	4.4.3
BRIEF { ON / OFF }	none	Set brief mode to print truncated (ON) or full (OFF) lines.	4.2.7
CHANGE	C	Replace, in the current line, the first occurrence of "string 1" with "string 2".	4.5.6
CLOSE	none	Terminate editing on input file.	4.6.1
CONVERT	none	Change contents of all lines in the file (or buffer) identified by a given text string (#1) to that of a second given text string (#2).	4.5.10
DELETE	D	Discard the current line.	4.5.2
EXIT	none	Transfer control to Monitor.	4.6.4
FIND	F	Bring first line beginning with "string" to work area.	4.4.4
GET	G	Add lines from subsidiary input device after (below) current line.	4.3.4
ICLOSE	E	Close, and transfer control to Monitor. (V5A GO TO Monitor.)	4.6.2
INSERT	I	Change mode to input.	4.5.4
INSERT string	I string	Add "string", as a complete line, to the file after (below) the current line.	4.5.5
KEEP	K	Keep (i.e., preserve) the original file (on. DAT-14) for backup purposes.	4.2.9
LC (Line Convert)	LC	Same as CONVERT command except that the number of lines to be converted is specified in the command.	4.5.11
LOCATE	L	Bring first line containing "string" to work area.	4.4.5

COMMAND	ABBREVIATION	OPERATION	PARAGRAPH REFERENCE
MODIFY	none	Search the current line for a specified text string; when it is found, accept new inputs from the keyboard and append them to the line starting after the last character of the given text string.	4.5.12
MOVE	none	Move a specified number of lines including the current line, to a position in the file (or buffer) immediately above a line identified by a given text string.	4.5.9
NEXT	N	Bring next consecutive line to work area.	4.4.2
OPEN	none	Prepare file on input device for editing.	4.2.3
OVERLAY	O	Replace multiple lines.	4.5.7
PRINT	P	Print the current line on the Teletype.	4.5.1
READ	none	Fill block buffer from input file.	4.3.1
RENEW	none	Perform buffer WRITE/READ functions.	4.3.3
RETYPE	R	Replace current line with "string".	4.5.3
SCLOSE	none	Close file and leave on output device.	4.6.3
SIZE	S	Set total lines to occupy block buffer.	4.2.3
TOP	T	Reset pointer to beginning of file.	4.4.1
VERIFY { ON / OFF }	V	Set VERIFY mode to print (ON) or ignore printing (OFF) lines after processing CHANGE, LOCATE, and FIND requests.	4.2.6
WRITE	none	Add block buffer to output file.	4.3.2

SPECIAL COMMANDS

CALL DELETE { INPUT / OUTPUT }	Delete named file from input or output device. (No abbreviation.)	4.2.2
CALL RENAME { INPUT / OUTPUT }	Rename named file located on specified input or output device. (No abbreviation.)	4.2.1

APPENDIX B

EDITVP

In PDP-15 ADSS or DOS systems which have a VP15A Storage Tube Display Unit in their hardware configuration, the user can call program EDITVP for use in editing operations in place of EDIT. The program EDITVP permits the storage tube to be used as a display device during editing operations. All standard editing commands (as described in Section 4) may be used in EDITVP operations with the following exceptions:

a) TV { ON / OFF — The TV ON command enables the operation of the display tube; this command implies VERIFY OFF (no console printer output). The command TV OFF disables the VP display operations; however, this command does not imply VERIFY ON.

During EDITVP editing operations, if any part of the current line is modified (as with RETYPE, CHANGE, OVERLAY or DELETE commands) a dashed line is drawn through the displayed line containing the change.

DISPLAY CONTROLS

The VP15A display unit is provided with two pushbutton controls, ERASE and VIEW, which are located on the right front panel of the CRT enclosure. These controls are used in the following manner:

a) ERASE — when operated, it causes the current display to be erased from the display.

b) VIEW — The normal brightness of displayed data fades after 90 seconds unless renewed; this control enables the user to renew (brighten) the display when desired.

WARNING

The same display should not be maintained on the CTR for a period exceeding 15 minutes; if this occurs, the phosphor of the display CRT may be damaged.

DISPLAY MODES

The VP15A CRT is capable of displaying 56, 72-character lines. If the data file to be displayed exceeds the 56-line display capacity, the display will operate in either a SCAN or PAGE mode, depending on the type of handler installed in the system (VPA or VPA.S) for theVP15A[1].

SCAN Mode

When the VPA device handler is installed in the operating system, the VP15A display operates in a SCAN mode. In this mode, each time the display screen is filled it is erased automatically after the 56th line is written and is refilled from the display fiee. This fill-erase-rewrite cycle is continued until the screen is completely or partially filled and no more data is in the file to be displayed. The last display is not erased but remains and may be renewed (brightened) manually by the viewer when necessary.

PAGE Mode

When the VPA.S handler is installed in the operating system, the VP15A display operates in a PAGE mode. In this mode, the erase-rewrite operations of the display are controlled by the settings of the unit ERASE control and the computer console switch ACSØ. With the ERASE and ACSØ controls set, each time the display is filled (i.e. full page) the display is not automatically erased and the next set of data displayed. This operation is not performed until the ERASE control is actuated by the user. This feature permits the user to manually advance through a file being edited, page by page.

EDITING OPERATIONS

The following notes describe some of the features of EDITVP and the manner in which it performs some of the basic editing functions.

---

[1]The type of VP15A handler available in the DOS operating system will be that selected by the System Manager during the configuration of the system.

a) Announcements, error messages and ">" symbols are echoed on the console printer, with one exception: the message "DAT SLOT xxx IS FILE ORIENTED", which depends on the entry state of ↑X (Control X).

b) Locative operations, the file lines passed over during locative operations are displayed as the file is scanned. The last line displayed is always the current line (found line).

c) CONVERT and LC command operations, only those lines changed in a file as the result of the command are displayed.

d) GET Command, the data to be added to a file from the subsidiary device is displayed on the screen as it is added to the current file or buffer.

e) DELETE Command, on the execution of a DELETE command, dashed lines are drawn through deleted items.

f) PRINT Command, the PRINT command implies "hard copy"; the line(s) selected by the command are output on the console printer. The current line is always the last line displayed.

g) Clear Screen Commands, the display screen is cleared by the entry of any of the following commands:

1) TOP
2) CLOSE
3) EXIT
4) TV OFF
5) Change the state of #6 pushbutton and screen will clear when the next output is made to the VT15 (e.g., new line added, mode change).

APPENDIX C

EDITVT

DOS systems which have a VT15 Graphics Display unit permit the user to employ program EDITVT for editing purposes. Program EDITVT enables the user to perform soft-copy editing of files using the VT15 display as a file data display device. Data is displayed in sets of either 56, 72-character lines or 28, 72-character lines. The EDITVT commands and the editing functions performed are essentially the same as those of the standard Editor Program (EDIT).

SETUP COMMANDS

The following command must be issued to the monitor prior to loading EDITVT:

a) $ VT ON — Enables the VT display unit.

b) $ HALF ON/OFF — This command is optional; it enables the user to set up a half-screen display (i.e., 28, 72-character lines) condition in which only half the screen is used for display.

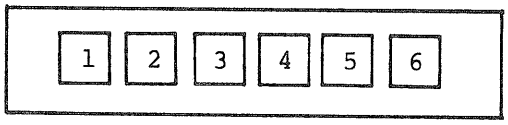
The program EDITVT is loaded into core by the command "EDITVT" given to the monitor. Once loaded, the program will announce itself by outputting its name and version number on the console printer. The user must then input the command "TV ON" to initiate the VT15 display operations. VT15 display operations may be stopped at any time by the command "TV OFF".

VT-15 CONTROLS AND DISPLAY MODES

CONTROLS

The VT-15 Display console contains a horizontal strip of six square push-to-light pushbuttons which are used in display operations. These pushbuttons are unmarked since their function is determined by software and may vary according to the particular program (system or user) which is in control of the system.

In EDIT operations, only the two rightmost pushbuttons are needed; these switches are referred to as #'s 5 and 6, based on the following numbering scheme:



DISPLAY MODES

The VT-15 Display operates in two display modes:

a) SCROLL Mode - When the #5 pushbutton is in the OFF (unlit) position, the display is in the SCROLL mode. In this mode, when the display screen is full, the next line of data to be displayed causes the displayed material to "roll" upwards, line-by-line, with new data displayed at the bottom of the screen.

b) PAGE Mode - When the #5 pushbutton is on the ON (lit) position, the display is in the PAGE mode. In this mode, when the screen is full, the next entered material for display causes the complete, full-screen display (i.e., PAGE) to be erased; the new material is then displayed starting at the top line of the screen.

WARNING

The EDITOR may appear hung when #5 is ON (lit) and the page is full (i.e., waiting for change in state of #6).

PAGING OPERATION

When a large file is to be displayed in the PAGE mode, the #6 control pushbutton is used to advance the display through the file page (screen) by page (screen). Each time this pushbutton is actuated, the screen is cleared and the next set of data available is displayed.

EDITING OPERATIONS

The following notes describe some of the features of EDITVT and the manner in which it performs some of the basic editing functions.

a) Announcements, error messages and ">" symbols are echoed on the console printer, with one exception: the message "DAT SLOT xxx IS FILE ORIENTED", which depends on the entry state of ↑X (Control X).

b) Locative operations, the file lines passed over during locative operations are displayed as the file is scanned. The last line displayed is always the current line (found line).

c) CONVERT and LC command operations, only those lines changed in a file as the result of the command are displayed.

d) GET Command, the data to be added to a file from the subsidiary device is displayed on the screen as it is added to the current file or buffer.

e) DELETE Command, on the execution of a DELETE command, the line(s) affected are erased from the screen and the remainder of the file is rolled up to fill the deleted (erased) line(s).

f) PRINT Command, the PRINT command implies "hard copy"; the line(s) selected by the command are output on the console printer. The current line is always the last line displayed.

g) Clear Screen Commands, the display screen is cleared by the entry of any of the following commands:

1) TOP
2) CLOSE
3) EXIT
4) TV OFF
5) Change the state of #6 pushbutton and screen will clear when the next output is made to the VT15 (e.g., new line added, mode change).

HOW TO OBTAIN SOFTWARE INFORMATION

Announcements for new and revised software, as well as programming notes, software problems, and documentation corrections are published by Software Information Service in the following newsletters.

Digital Software News for the PDP-8 & PDP-12
Digital Software News for the PDP-11
Digital Software News for the PDP-9/15 Family

These newsletters contain information applicable to software available from Digital's Program Library, Articles in Digital Software News update the cumulative Software Performance Summary which is contained in each basic kit of system software for new computers. To assure that the monthly Digital Software News is sent to the appropriate software contact at your installation, please check with the Software Specialist or Sales Engineer at your nearest Digital office.

Questions or problems concerning Digital's Software should be reported to the Software Specialist. In cases where no Software Specialist is available, please send a Software Performance Report form with details of the problem to:

Software Information Service
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

These forms which are provided in the software kit should be fully filled out and accompanied by teletype output as well as listings or tapes of the user program to facilitate a complete investigation. An answer will be sent to the individual and appropriate topics of general interest will be printed in the newsletter.

Orders for new and revised software and manuals, additional Software Performance Report forms, and software price lists should be directed to the nearest Digital Field office or representative. U.S.A. customers may order directly from the Program Library in Maynard. When ordering, include the code number and a brief description of the software requested.

Digital Equipment Computer Users Society (DECUS) maintains a user library and publishes a catalog of programs as well as the DECUSCOPE magazine for its members and non-members who request it. For further information please write to:

DECUS
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

READER'S COMMENTS

EDITOR Utility Manual
DEC-15-YWZB-DN6

Digital Equipment Corporation maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback -- your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy. organization, usability and readability.

Did you find errors in this manual? If so, specify by page.

How can this manual be improved?

Other comments?

Please state your position. _____ Date: _____

Name: _____ Organization: _____

Street: _____ Department: _____

City: _____ State: _____ Zip or Country _____

— — — — — — — — — — — — — — — — Fold Here — — — — — — — — — — — — — — — — — —

— — — — — — — — — — — — Do Not Tear - Fold Here and Staple — — — — — — — — — — — —

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

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

Postage will be paid by:

digital

Digital Equipment Corporation
Software Information Services
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754