

**EDIT/EDITVP/EDITVT
XVM UTILITY MANUAL**

DEC-XV-UETUA-A-D



**XVM
Systems**
digital

**EDIT/EDITVP/EDITVT
XVM UTILITY MANUAL**

DEC-XV-UETUA-A-D

First Printing, December 1975

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by DIGITAL.

Copyright (C) 1975 by Digital Equipment Corporation

The postage prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-10
DECCOMM		TYPESET-11

CONTENTS

		Page
PREFACE		vii
CHAPTER 1	INTRODUCTION	1-1
CHAPTER 2	FUNCTIONAL DESCRIPTION	2-1
2.1	CONTROL MODES	2-1
2.2	DATA MODES	2-1
2.2.1	Line-by-Line Data Mode	2-1
2.2.2	Block Data Mode	2-2
2.3	DATA FILES	2-3
2.3.1	Using Monitor I/O	2-3
2.3.2	Input and Subsidiary Files	2-4
2.3.3	Output Files	2-5
2.4	USE OF THE CTRL P KEYBOARD COMMAND	2-6
2.5	CHARACTER ERASE AND LINE DELETE KEYBOARD COMMANDS	2-6
CHAPTER 3	EDITING OPERATIONS	3-1
3.1	MODIFYING AN EXISTING FILE	3-1
3.2	INPUT/EDIT MODES	3-2
3.3	CREATING A NEW FILE	3-2
3.4	EDIT BLOCK MODE	3-3
3.5	CLOSING THE NEW FILE	3-3
3.6	ERROR-HANDLING CONVENTIONS	3-4
3.6.1	Command String Errors	3-4
3.6.2	Premature End-of-File	3-4
3.6.3	Read Errors and Line Overflow	3-5
3.6.4	Block Mode Buffer Overflow	3-6
3.6.5	File Naming and Calling Errors	3-6
3.6.5.1	Absent File	3-6
3.6.5.2	Absent File Name	3-7
3.6.5.3	Identically Named Files	3-7
3.6.5.4	Nothing in File	3-8
3.7	FILE RENAMING AND DELETION	3-8
CHAPTER 4	EDIT MODE COMMANDS	4-1
4.1	INTRODUCTION	4-1
4.1.1	Example File	4-2
4.2	SETUP COMMANDS	4-2
4.2.1	CALL RENAME Command	4-2
4.2.2	CALL DELETE Command	4-3
4.2.3	OPEN Command	4-4
4.2.4	PUT Command	4-5
4.2.5	BLOCK ON/OFF Command	4-5
4.2.6	SIZE [S] Command	4-7
4.2.7	VERIFY [V] ON/OFF Command	4-8
4.2.8	BRIEF ON/OFF Command	4-9
4.2.9	OUTPUT ON/OFF Command	4-10
4.2.10	KEEP [K] Command	4-12
4.2.11	PARITY ON/OFF Command	4-13
4.3	INPUT/OUTPUT COMMANDS	4-13
4.3.1	READ Command	4-14

CONTENTS (Cont.)

		Page
4.3.2	WRITE Command	4-15
4.3.3	RENEW Command	4-15
4.3.4	GET [G] Command	4-16
4.4	LINE POINTER CONTROL (LOCATIVE) COMMANDS	4-17
4.4.1	TOP [T] Command	4-18
4.4.2	NEXT [N] Command	4-18
4.4.3	BOTTOM [B] Command	4-20
4.4.4	FIND [F] Command	4-20
4.4.5	LOCATE [L] Command	4-21
4.5	TEXT MODIFICATION AND MANIPULATIVE COMMANDS	4-22
4.5.1	PRINT [P] Command	4-22
4.5.2	LIST Command	4-24
4.5.3	DELETE [D] Command	4-25
4.5.4	RETYPE [R] Command	4-26
4.5.5	INSERT [I] Command	4-27
4.5.6	INSERT [I] Line Command	4-28
4.5.7	CHANGE [C] Command	4-29
4.5.8	OVERLAY [O] Command	4-30
4.5.9	APPEND [A] Command	4-31
4.5.10	MOVE Command (DOS SYSTEMS ONLY)	4-32
4.5.11	CONVERT Command (DOS SYSTEMS ONLY)	4-36
4.5.12	LC (Line Convert) Command (DOS SYSTEMS ONLY)	4-38
4.5.13	MODIFY [M] Command (DOS SYSTEMS ONLY)	4-40
4.6	EDIT CLOSE OPERATION COMMANDS	4-42
4.6.1	CLOSE Command	4-42
4.6.2	PCLOSE Command	4-42
4.6.3	SCLOSE Command	4-43
4.6.4	EXIT [E] Command	4-43
APPENDIX A	SUMMARY OF EDITING COMMANDS	A-1
APPENDIX B	EDITVP	B-1
APPENDIX C	EDITVT	C-1
INDEX		INDEX-1

FIGURES

		Page
Figure 2-1	Schematic of Line Processing in Block and Normal Modes	2-3

TABLES

		Page
Table 2-1	Standard DAT and UFDI Slot Assignments for the XVM/DOS Monitor	2-4
2-2	Output File Conventions for the Text Editor	2-6

LIST OF ALL XVM MANUALS

The following is a list of all XVM manuals and their DEC numbers, including the latest version available. Within this manual, other XVM manuals are referenced by title only. Refer to this list for the DEC numbers of these referenced manuals.

BOSS XVM USER'S MANUAL	DEC-XV-OBUAA-A-D
CHAIN XVM/EXECUTE XVM UTILITY MANUAL	DEC-XV-UCHNA-A-D
DDT XVM UTILITY MANUAL	DEC-XV-UDDTA-A-D
EDIT/EDITVP/EDITVT XVM UTILITY MANUAL	DEC-XV-UETUA-A-D
STRAN XVM UTILITY MANUAL	DEC-XV-UTRNA-A-D
FOCAL XVM LANGUAGE MANUAL	DEC-XV-LFLGA-A-D
FORTRAN IV XVM LANGUAGE MANUAL	DEC-XV-LF4MA-A-D
FORTRAN IV XVM OPERATING ENVIRONMENT MANUAL	DEC-XV-LF4EA-A-D
LINKING LOADER XVM UTILITY MANUAL	DEC-XV-ULLUA-A-D
MAC11 XVM ASSEMBLER LANGUAGE MANUAL	DEC-XV-LMLAA-A-D
MACRO XVM ASSEMBLER LANGUAGE MANUAL	DEC-XV-LMALA-A-D
MTDUMP XVM UTILITY MANUAL	DEC-XV-UMTUA-A-D
PATCH XVM UTILITY MANUAL	DEC-XV-UPUMA-A-D
PIP XVM UTILITY MANUAL	DEC-XV-UPPUA-A-D
SGEN XVM UTILITY MANUAL	DEC-XV-USUTA-A-D
SRCCOM XVM UTILITY MANUAL	DEC-XV-USRCA-A-D
UPDATE XVM UTILITY MANUAL	DEC-XV-UUPDA-A-D
VP15A XVM GRAPHICS SOFTWARE MANUAL	DEC-XV-GVPAA-A-D
VT15 XVM GRAPHICS SOFTWARE MANUAL	DEC-XV-GVTAA-A-D
XVM/DOS KEYBOARD COMMAND GUIDE	DEC-XV-ODKBA-A-D
XVM/DOS READER'S GUIDE AND MASTER INDEX	DEC-XV-ODGIA-A-D
XVM/DOS SYSTEM MANUAL	DEC-XV-ODSAA-A-D
XVM/DOS USERS MANUAL	DEC-XV-ODMAA-A-D
XVM/DOS VIA SYSTEM INSTALLATION GUIDE	DEC-XV-ODSIA-A-D
XVM/RSX SYSTEM MANUAL	DEC-XV-IRSMA-A-D
XVM UNICHANNEL SOFTWARE MANUAL	DEC-XV-XUSMA-A-D

Handwritten marks or characters along the right edge of the page, possibly bleed-through from the reverse side.

PREFACE

The operation and use of the EDIT/EDITVP/EDITVT XVM Utility program (EDIT) are described in this manual.

It was assumed in the preparation of this manual that the reader is familiar with the operation of the XVM or PDP-15 equipment and the contents of the software manual describing the features of XVM/DOS (XVM/DOS Software System User's Manual).

This manual also serves as complementary documentation for the XVM/RSX EDITOR. However, the description of DAT slots given in this manual does not apply to the XVM/RSX EDITOR.

CHAPTER 1 INTRODUCTION

The EDIT/EDITVP/EDITVT XVM Utility program is a powerful context-editing program that allows the modification and creation of symbolic source programs and other ASCII text material¹. 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 with the Disk Operating (XVM/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.

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.

¹The Editor reads and writes standard IOPS ASCII lines. The characteristics of IOPS ASCII text are described in the XVM/DOS User's Manual.

Introduction

A summary of the standard Editor commands is presented in Appendix A. Special symbols, when used in this manual, are defined as follows:

<u>Symbols</u>	<u>Represents</u>
↵	Carriage RETURN-LINE FEED operation
→	TAB
␣	SPACE
[]	Optional Command String elements

CHAPTER 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 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

Functional Description

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 portion¹ 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.

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).

¹If not specified, a group of 55₁₀ lines is assumed (default size) by the Editor.

Functional Description

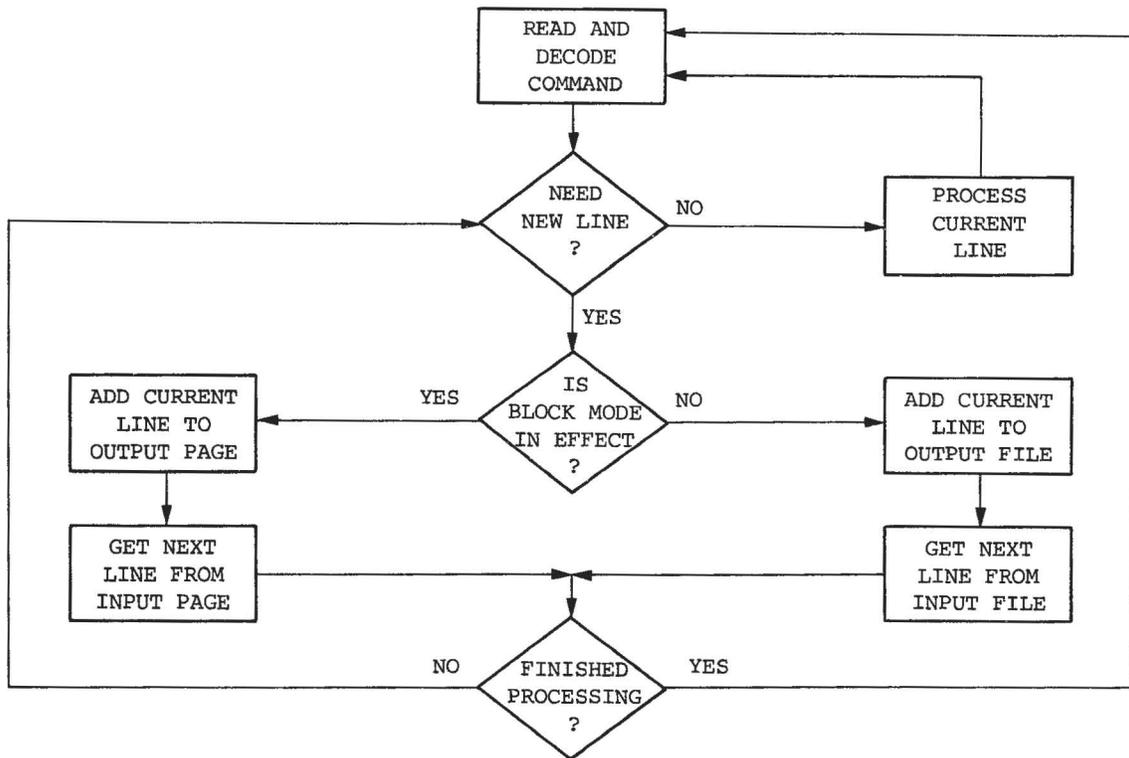


Figure 2-1
Schematic of Line Processing in
Block and Normal Modes

2.3 DATA FILES

The following paragraphs describe the XVM/DOS 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 the XVM/DOS software environment, the Editor makes use of the monitor's Input/Output Programming System (IOPS) for I/O transfer operations. The Editor communicates with IOPS via separate Device Assignment (DAT) and User File Directory (UFDT) tables. Refer to the XVM/DOS User's Manual for a description of these tables and their use.

Functional Description

Table 2-1
Standard DAT and UFD Slot Assignments for the XVM/DOS Monitor

.DAT	DEVICE	UIC(1)	USE
-15	DKA	GEP ¹	Output/Scratch
-14	DKA	GEP	I/O
-13	DKA	GEP	Secondary Output
-12	LPA	GEP	Listing Output
-10	TTA	GEP	Secondary Input
+10	VPA	GEP	(EDITVP only)

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.

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 (.DAT -14),
- c) a subsidiary input device/UFD which contains, normally, prepared material to be inserted into the object file (.DAT -10).

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.

The subsidiary input device may be either directoried or non-directoried. When the GET command is typed, the Editor issues a .FSTAT to .DAT -10. If the device is directoried, the entire contents of the file named in the GET command is inserted into the output file. If, however, the device is non-directoried, "n" lines will be inserted into the file.

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

Functional Description

2.3.3 Output Files

The Editor, upon receiving control, attempts to determine if the input and scratch devices are directoried. If either one of the devices is not directoried, then the scratch device (.DAT -15) is assigned as the final output device. If both devices are directoried, the scratch device is assigned an intermediate function and the input device is used as the final output device.

If, in XVM/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 directoried), 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 directoried, the newly 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

Functional Description

the new file has been successfully written and closed. A system failure, therefore, can never result in total loss of data.

Table 2-2
Output File Conventions for the Text Editor

Input Device	Scratch or Output Device	Edited File Appears on:	Input File is:
Directoried	Directoried	Input Device	Deleted
Directoried	Non-directoried	Output Device	Unchanged
Non-directoried	Directoried	Output Device	Unchanged
Non-directoried	Non-directoried	Output Device	Unchanged

In addition, it is possible, through use of the PUT and PCLOSE commands, to create alternate output files which will reside on the device assigned to .DAT -13. The use of these commands is discussed fully in Sections 4.2.4 and 4.6.2, respectively.

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:



CHAPTER 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 XVM Vnxnnn
```

where:

nx = a version number

and

nnn = an edit number

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

3.1 MODIFYING AN EXISTING FILE

If the input device has a directory (disk, magnetic tape, or DECTape), the first command to the Editor must be:

```
OPEN filename ext ↵
```

where "filename" is the primary name¹ of the wanted file residing on the input device and "ext" is its extension². 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

¹Maximum of six characters permitted for "filename".

²Maximum of three characters permitted for "ext".

Editing Operations

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.

If the input device is not directoried (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 non-directoried 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 or ALT MODE 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 directoried, 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.

Editing Operations

If the output device is directoried, 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 directoried, 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 Section 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 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 Section 4.2.5). If not specified, the Editor assumes a default size of 55₁₀ 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 Section 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 Section 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.

Editing Operations

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".
- c. A command was issued in an illogical place, OPEN following an OPEN command.

When the BRIEF Mode (see Section 4.2.8) 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

Editing Operations

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 below the final line of the group being read.

3.6.3 Read Errors and Line Overflow

During DOS and RSX editing operations, inputs from either the input or subsidiary file are checked for two possible errors. 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

Editing Operations

of more than 135₁₀ characters plus a terminator (DOS) or 80₁₀ characters plus a terminator (RSX) results in the comment:

TRUNCATED:

followed by the first (leftmost) 135 (or 80₁₀) 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)

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.

Editing Operations

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 like-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:

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.

If the file processing has proceeded to a point at which recovery, as described above, is impossible, the Editor recognizes a priority

Editing Operations

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 .TFILL 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 -14 (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 a CLOSE command prior to a 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, control returns to the Editor. The contents of the buffer remain unchanged. In the case of directoried 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 Chapter 4.

CHAPTER 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 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␣)
```

Edit Mode Commands

4.1.1 Example File

The following file or portions of it are used throughout this chapter 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 Sections 4.2.1, through 4.2.9.

4.2.1 CALL RENAME Command

Form:

$$\text{CALL} \text{ _ } \text{RENAME} \text{ _ } \left\{ \begin{array}{l} \text{INPUT} \\ \text{OUTPUT} \end{array} \right\} \text{ _ } \text{OLDNAM} \text{ _ } \text{EXT} \text{ _ } \text{NEWNAM} \text{ _ } \text{EXT}$$

Description:

This command must be used before the OPEN command (see 4.2.3) is issued or after a CLOSE command. 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.

Edit Mode Commands

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.

```
24-NOV-75                                     listing of current UFD in
DIRECTOR Y LISTING (VIC)                       DOS System
 2436 FREE BLKS
   1 USER FILES
   1 USER BLKS
XXX   SPC           1  24-NOV-75               entry for file "XXX SRC"

EDITOR XVN Vxxxxxx
>CALL RENAME INPUT XXX SRC NMTST SRC          RENAME Command, change name
>                                              of file XXX SRC to NMTST SRC

24-NOV-75                                     listing of resultant UFD
DIRECTOR Y 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 }        FILNAM EXT       
                     { OUTPUT }
```

Description:

This command must be issued before the OPEN command (see 4.2.3) is issued or after a CLOSE command. 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.

```
24-NOV-75                                     listing of current UFD in DOS
DIRECTOR Y LISTING (VIC)                       System
 2436 FREE BLKS
   1 USER FILES
   1 USER BLKS
NMTST SRC           1  02-OCT-71              entry for file "NMTST SRC"

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

24-NOV-75                                     listing of resultant UFD
DIRECTOR Y LISTING (VIC)
 2437 FREE BLKS
   0 USER FILES
   0 USER BLKS
```

Edit Mode Commands

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 filename ext 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 nonexistent file are named in an OPEN command.

```
EDITOR XVN Vxnnn  
>OPEN EDTST SRC  
EDIT  
>
```

} *opening an existing file*

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

} *OPEN Command referencing
a nonexistent file*

Edit Mode Commands

4.2.4 PUT Command

Form:

PUT filename ext ↵

Description:

Frequently, the user wishes to place portions of a file into many separate source files. The PUT command allows the user to do this. If the user specifies a filename ext that already exists, the program will type:

```
filename ext IS ALREADY PRESENT ON .DAT -13.
```

and will then wait for the next Editor command.

If no file name is used, the input file name is assumed with an ext of SRC and the program types:

```
NO FILE NAME SPECIFIED:  
FILE: filename ext IS ASSUMED.
```

While the PUT is in effect, the output stream will be copied to .DAT -13.

4.2.5 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 55₁₀ lines may be accepted.

Edit Mode Commands

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-directoryed device, the Block Mode is automatically turned ON by Edit. If both input and scratch devices are directoryed 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, load and print out the buffer:

```
↑P
EDITOR XVN Unknown
>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
                   /EDIT COMMAND DEMONSTRATOR
TAG1  .TITLE  EXAMP
      AAA    /LINE 1
      BBB
      CCC
>
```

CONTENTS OF BLOCK MODE BUFFER

Edit Mode Commands

- 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
>BLOCK OFF
>T
>P 20
TAG1  .IITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
      AAA    /LINE 1
      BBB
      CCC
TAG2  DDD     /LINE 4
      FEE
      FFF
TAG3  GGG     /LINE 7
      HHH
      III
      .END
END OF FILE REACHED BY:
P 20
>
```

empty buffer
turn BLOCK mode OFF
go to the top of the file in core
PRINT up to 20 lines

↑
printout of file loaded in core for Edit operations
↓

message printed to indicate that the end of the file was reached before 20 lines were printed.

4.2.6 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 3 2).

Description:

This command applies only to READ and RENEW 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.

Edit Mode Commands

If a SIZE command is not used, the Editor program assumes a default size of 55₁₀ 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) S₅ establishes a 5-line buffer
- b) S₁₀₀ establishes a 100-line buffer

4.2.7 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.

Edit Mode Commands

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	} current contents of block buffer
	BBB		
	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.

4.2.8 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.

Edit Mode Commands

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 (i.e., 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
>I                 go to Top of buffer
>F TAG             find line containing the string "TAG"
TAG1   AAA        total line is printed to Verify operation
                /LINE 1
>I                 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
```

4.2.9 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 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.

Edit Mode Commands

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.

<pre> ↑P EDITOR XVN Vnxnrrr >OPEN EXAMP SRC EDIT >P 12 TAG1 .TITLE EXAMP /EDIT COMMAND DEMONSTRATOR AAA /LINE 1 BBB CCC TAG2 DDD /LINE 4 EEE FFF TAG3 GGG /LINE 7 HHH III .END >I >F TAG2 TAG2 DDD /LINE 4 >OUTPUT OFF >F TAG3 TAG3 GGG /LINE 7 >OUTPUT ON >R .END >CLOSE </pre>	<p><i>open demonstration program</i></p> <p><i>Print 12 lines of current file</i></p> <hr/> <p><i>current (original) contents of file being edited.</i></p> <hr/> <p><i>go to Top of file</i> <i>Find line starting with "TAG2"</i> <i>printout of selected line</i> <i>OUTPUT "OFF" command</i> <i>Find line starting with "TAG3"</i> <i>printout of selected line</i> <i>OUTPUT "ON" command</i> <i>go to Bottom of file</i> <i>printout of last line in file</i> <i>Close file and terminate Edit operations</i></p>
<pre> EDITOR XVN Vnxnrrr >OPEN EXAMP SRC EDIT >P 12 TAG1 .TITLE EXAMP /EDIT COMMAND DEMONSTRATOR AAA /LINE 1 BBB CCC TAG3 GGG /LINE 7 HHH III .END END OF FILE REACHED BY: P 12 > </pre>	<p><i>reopen demonstrator program</i></p> <p><i>Print 12 lines</i></p> <hr/> <p><i>contents of the demonstrator program as modified by the OUTPUT ON/OFF command in the previous Edit operation.</i></p> <hr/>

Edit Mode Commands

```
EDITOR XVN Vnxxxx
>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 XVN Vnxxxx
>
```

```
24-NOV-75
DIRECTORY LISTING (JAN)
2436 FREE BLKS
2 USER FILES
2 USER BLKS
EXAMP2 SRC      1 24-NOV-75 -- original file
EXAMP SRC       1 24-NOV-75 -- new file
```

*} listing of UFD on
input .DAT slot after
Edit operations*

4.2.11 PARITY ON/OFF Command

Form:

PARITY $\left\{ \begin{array}{l} \text{ON} \\ \text{OFF} \end{array} \right\}$

Description:

When the Editor is initially loaded and restarted, PARITY is enabled. This means that paper tapes which are input will be parity checked. However, parity checking can be turned off via this command to enable reading of paper tapes with no parity. In this case, the user must take special care to be sure that the data input is correct.

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.

Edit Mode Commands

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 55₁₀ is used.

On 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 XUN Vrxnnn
>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
TAG1      .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
          AAA     /LINE 1
          BBB
          CCC
          contents of
          block buffer
>WRITE
>READ
>P 5
TAG2      DDD     /LINE 4
          FEE
          FFF
          GGG     /LINE 7
          contents of
          block buffer
>
```

Edit Mode Commands

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

Example:

```
EDITOR XUN Vnxxxxn
>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
TAG1   .TITLE   EXAMP  /EDIT COMMAND DEMONSTRATOR
      AAA      /LINE 1
      BBB
      CCC      } contents of buffer
>RENEW           write, clear and fill buffer
>P 5                PRINT 5 lines
TAG2   DDD      /LINE 4
      EEE
      FFF      } new contents of buffer
TAG3   GGG      /LINE 7
>
```

Edit Mode Commands

4.3.4 GET [G] Command

Form:

$$\text{GET } \left\{ \begin{array}{l} n \\ \text{filename ext} \end{array} \right\}$$

where:

n is a decimal number

or

filename_ext is the name of the file to be inserted

NOTE

GET may be abbreviated as G (e.g., G_3_).

Description:

In general, this command causes some number of 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 last line read from the subsidiary device is left in the work area as the current line.

If the device assigned to .DAT -10 (subsidiary input) is non-directoryed, the GET command expects the argument "n" which is the decimal number of lines to be read. 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.

If a directoryed device¹ is assigned to .DAT -10, the GET command expects the argument "filename_ext". The entire contents of the named file will be inserted into the output file.

¹Not supported in the XVM/RSX EDITOR.

Edit Mode Commands

Example:

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

```
EDITOR XUN Unxxxx
>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
THIS MESSAGE IS BEING ADDED TO THE OUTPUT FILE
FROM THE SUBSIDIARY INPUT DEVICE (PR IN THIS CASE)
USING THE GET COMMAND
        EEE
        FFF
TAG3    GGG      /LINE 7
        HHH
        III
        .END
END OF FILE REACHED BY:
P 20
>
```

inserted lines } *contents of file*

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.

Edit Mode Commands

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 XVN Vnxxxx
>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 5)

Description:

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

Edit Mode Commands

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. The pointer is positioned past the last line of the file. More text can be added at this point and all editor commands are legal.

Example:

The following illustrates the use of the NEXT command.

```
EDITOR XVN Vnxxxx
>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
>
```

Edit Mode Commands

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 BOTTOM command will move the pointer to the last line of a buffer:

```
↑P
EDITOR XVN Vnxonn
>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)
TAG1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
      AAA     /LINE 1
      BBB
      CCC
TAG2  DDD     /LINE 4
>T
>BOTTOM            move pointer to TOP of buffer
TAG2  DDD     /LINE 4  move pointer to last line of buffer
>                 line printed to verify operation
```

} current contents
of buffer

4.4.4 FIND [F] Command

Form:

FIND_↵string_↵

NOTE

This command may be abbreviated as F
(e.g., F_↵ABC_↵).

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

Edit Mode Commands

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 } REACHED BY:  
      { BUFFER }  
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 FIND command.

```
EDITOR XVN Vnkxnan  
>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., LABC).

Description:

This command causes a 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 consist of only printable characters and need not be a complete word. The string given, however, should be unique, within the file or buffer, to the desired line.

Edit Mode Commands

The initiated search starts at the next line. If the specified string is found, the line containing 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 $\left. \begin{array}{c} \text{FILE} \\ \text{BUFFER} \end{array} \right\}$ REACHED BY:

LOCATE (or L) string

is printed at the console.

Example:

The following illustrates the use of the LOCATE command:

```
EDITOR XVN VXXXXXXXX
>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 4.5.13. Explicit examples of the use of this type of Edit command are given wherever possible.

4.5.1 PRINT [P] Command

Form:

PRINT $\underline{\quad}$ n) where n is a decimal number.

NOTE

This command may be abbreviated as P
(e.g., P $\underline{\quad}$ n).

Edit Mode Commands

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 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 PRINT command.

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

Edit Mode Commands

4.5.2 LIST Command¹

Form:

LIST n where n is a decimal number

Description:

This command permits the user to list to the device (generally a line printer) assigned to .DAT -l2, one line or a group of "n" lines of the current file or buffer. List 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 one. The command LIST causes the current line to be printed; however, it will also remain the "current" line after being printed.

After a 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 } REACHED BY:  
      { BUFFER }  
LIST n
```

is printed at the console.

Because of the possibility of listing to a UNICHANNEL line printer with spooling enabled, the EDITOR issues a .CLOSE command at the end of each operation and initializes the printer handler to not perform a form feed when .CLOSE is received.

¹Not supported by the XVM/RSX editor.

Edit Mode Commands

```
>T                                go to TOP of buffer
>L CCC                            LOCATE line containing "CCC"
                                verification printout
                                CCC
                                DELETE 3 lines including current line
>D 3
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
END OF BUFFER REACHED BY:
P 6
>
```

} contents of
buffer after
DELETE operation

4.5.4 RETYPE [R] Command

Form:

RETYPE line)

NOTE

This command may be abbreviated as R
(e.g., R THIS IS A NEW 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 RETYPE command:

```
EDITOR KVN Unxxxxx
>OPEN EXAMP SRC                OPEN demonstration program
EDIT                            PRINT six lines
>P 6
                                .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
TAG1  AAA  /LINE 1
      BBB
      CCC
TAG2  DDD  /LINE 4
>T                                go to TOP of file
>L BBB                            LOCATE line containing "BBB"
                                verification printout
                                BBB
                                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
THIS MESSAGE REPLACES LINE BBB
      CCC
TAG2  DDD  /LINE 4
>
```

} first six lines
of file

} first five lines
of file; note
change in line
"BBB"

Edit Mode Commands

4.5.5 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 INSERT 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 null line (i.e., a line containing only a ↵). Once EDIT mode is restored, the current line pointer will be positioned at the last line entered.

Example:

The following illustrates the use of the INSERT command.

```
EDITOR XVN Vnxxxx
>OPEN EXAMP SRC          OPEN demonstrator program
EDIT
>P 6                     PRINT six lines
TAG1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
      AAA    /LINE 1
      BBB
      CCC
TAG2  DDD    /LINE 4
>I
>L BBB                  TOP
                        LOCATE line containing "BBB"
                        BBB
>INSERT                 INSERT command
INPUT                   INPUT mode identifier
1111  ADDED LINE
2222  ADDED LINE
3333  ADDED LINE
EDIT
>I                       EDIT mode identifier
>P 10                   TOP
                        PRINT ten lines
TAG1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
      AAA    /LINE 1
      BBB
      1111  ADDED LINE
      2222  ADDED LINE
      3333  ADDED LINE
      CCC
TAG2  DDD    /LINE 4
      EEE
>
```

} first five lines
of current file

} inserted material

} contents of file;
note inserted
material

Edit Mode Commands

4.5.6 INSERT [I] Line Command

Form:

INSERT, line)

NOTE

This command may be abbreviated as I
(e.g., I, THIS IS NEW.).

Description:

This command enables the user to insert a single line between lines of the 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 argument of the INSERT command becomes the current line. Note that the insertion is made below the current line when the command is issued. The EDITOR remains in the Edit Mode throughout the execution of this command.

Example:

The following illustrates the use of the INSERT command.

```
EDITOR XVN VXXXXXXXXX
>OPEN EXAMP SRC          OPEN demonstrator program
EDIT
>P 6                     PRINT six lines
TAG1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
      AAA    /LINE 1
      BBB
      CCC
TAG2  DDD    /LINE 4
>I                                     TOP
>L BBB                               LOCATE line "BBB"
      BBB
>INSERT 1111 INSERTED LINE  INSERT command
>I                                     TOP
>P 10
TAG1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
      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 line inserted immediately following line "BBB"

Edit Mode Commands

4.5.7 CHANGE [C] Command

Form:

CHANGE /string 1/string 2/↵

NOTE

This command may be abbreviated as "C" (e.g., C /KAT/CAT/↵); 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 the delimiter.

Description:

This command enables the user to replace selected portions of a current line. Part (string 1) 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 none.

Null strings to be replaced or entered are indicated by two consecutive separators such as //.

If the number of characters in the current line, including the terminator, exceeds 135₁₀ as the result of a CHANGE command, the message:

TRUNCATED

is printed at the console and the excessive right-hand characters (those exceeding 135₁₀) of the current line are discarded.

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

Edit Mode Commands

Example :

```
EDITOR XVN Unxxxx
>OPEN EXAMP SRC          OPEN demonstration program
EDIT
>P 4                     PRINT four lines
TAG1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR } first three
      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"
>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
ADD1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATION } first three
TAGB  AAA    /LINE 1 } lines of modified
      BBB                                     } file
>T                          TOP
>L TAGB                  LOCATE line containing "TAGB"
TAGB  BBB
>C /TAGB//              use null string to delete "TAGB"
      BBB
>
```

4.5.8 OVERLAY [O] Command

Form:

OVERLAY n) where n is a decimal number.

NOTE

This command may be abbreviated as O (e.g., O1).

Description:

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

Edit Mode Commands

The user may then enter any number of lines as replacements. The INPUT mode is terminated and the EDIT mode restored by the entry of a null line (i.e., a line containing only a `)`).

Example:

The following illustrates how the OVERLAY command is used to replace one line with two lines input from the console keyboard.

```
EDITOR XVN Vnxxxx
>OPEN EXAMP SRC          OPEN demonstrator program
EDIT
>P 5                     PRINT five lines
TAG1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
      AAA    /LINE 1
      BBB
      CCC                } first four lines
                          } of current file
>T
>L CCC                  LOCATE line "CCC"
      CCC
>OVERLAY                OVERLAY line "CCC"
INPUT                   INPUT mode identifier
THIS INPUT LINE IS TO REPLACE (OVERLAY) LINE CCC
THIS LINE IS TO BE INSERTED BETWEEN NEW LINE AND TAG2 } keyed in
                                                         } lines
EDIT                   restore EDIT mode
>T
>P 8
TAG1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
      AAA    /LINE 1
      BBB
THIS INPUT LINE IS TO REPLACE (OVERLAY) LINE CCC
THIS LINE IS TO BE INSERTED BETWEEN NEW LINE AND TAG2 } first six lines
TAG2  DDD    /LINE 4
      EEE                } of current file
                          } showing the
                          } input overlays
>
```

4.5.9 APPEND [A] Command

Form:

```
APPEND string)
```

NOTE

This command may be abbreviated as A
(e.g., Astring).

Description:

The APPEND command enables the user to add (append) data after the last character but before the line terminator of the current line and may contain any character.

Edit Mode Commands

Example:

The following illustrates the use of the APPEND command:

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

4.5.10 MOVE Command

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 which begins with "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.

Edit Mode Commands

The string specified 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 string (as in the FIND command).

Example:

The following example illustrates the use of the MOVE command.

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

Edit Mode Commands

Example:

```
↑P
EDITOR XUN Vnxnnn
>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 than are in the block buffer, the Editor 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 XUN Vnxnnn
>OPEN EXAMP SRC
EDIT
>L BBB
      BBB   /LINE 2
>MOVE 15 TAG3
NOT ENOUGH LINES
END OF FILE REACHED BY:
MOVE 15 TAG3
>
```

Edit Mode Commands

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 locative command.

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

```
      LINES MOVED TO END  
      END OF { BUFFER } REACHED BY:  
             { FILE }  
      MOVE _n_ string
```

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 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.

Edit Mode Commands

```
EDITOR XVN Vnxnnn
>OPEN EXAMP SRC
EDIT
>L TAG1
TAG1   AAA   /LINE 1
>MOVE 1 BBB
LINES MOVED TO END
END OF FILE REACHED BY:
MOVE 1 BBB
>I
>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
>I
>L BBB
      BBB
>MOVE 9 ZZZ
LINES MOVED TO END
END OF FILE REACHED BY:
MOVE 9 ZZZ
>I
>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
```

bad command; tab before "BBB" not included

resulting error message

resulting position of line "TAG1"

MOVE lines "BBB" through ".END" to end of file, i.e., below line "TAG1"

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 entered during the processing of the MOVE command is ignored. After the MOVE has been processed then a new CTRL P will be honored.

4.5.11 CONVERT Command

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.

Edit Mode Commands

Description:

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

This command scans the entire buffer or file and each time that it finds a string which matches string 1 it replaces that string with string 2 (as if a CHANGE command had been executed).

If a string 2 is not given in the CONVERT command, all strings matching 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, as if an End Of File/Buffer condition had occurred.

If, during the convert process, a line exceeds 135₁₀ characters including the terminator, it is truncated and the Editor outputs:

```
(truncated line)
TRUNCATED
```

and the convert process continues.

String 1 must be non-null, otherwise the message:

```
NO STRING
```

is output and the Editor waits for the next user command.

The following example illustrates the use of the CONVERT command.

```
EDITOR XVM VNXNNN
>OPEN EXAMP SRC
EDIT
>P 12
TAG1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
      AAA    /LINE 1
      BBB
      CCC
TAG2  DDD    /LINE 4
      EEE
      FFF
TAG3  GGG    /LINE 7
      HHH
      III
      .END
```

} original file

Edit Mode Commands

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

} converted file

If string 1 is never found, then the message

NOT FOUND

is output and the Editor waits for the next user command.

4.5.12 LC (Line Convert) Command

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 4.5.11); 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.

Edit Mode Commands

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

*original
file*

*converted
file*

Edit Mode Commands

4.5.13 MODIFY [M] Command

Form:

MODIFY string) or ALT MODE

NOTE

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

Description:

This command causes the contents of the current line up to the last character of the specified 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 which specifies that the remainder (unprinted portion) of the original line is to be added to the modified line;
- b) a CARRIAGE RETURN () which specifies that the unprinted portion of the original line is to be deleted.

A line may contain a maximum of 135₁₀ characters (including terminator); when this limit is reached during a MODIFY operation, the Editor issues a CARRIAGE RETURN and waits for the next user command. Care must be taken to ensure that the final line does not exceed 135₁₀ 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 data and the operation terminated by ALT MODE to restore the original line.

Edit Mode Commands

Example:

```
EDITOR XVN Vxxxxx
>OPEN EXAMP SRC          OPEN demonstrator program
EDIT
>P 12
TAG1  .TITLE  EXAMP  /EDIT COMMAND DEMONSTRATOR
      AAA    /LINE 1
      BBB
      CCC
TAG2  DDD    /LINE 4
      EEE
      FFF
TAG3  GGG    /LINE 7
      HHH
      III
      .END

```

} original file

```

>I
>L TAG2
TAG2  DDD    /LINE 4
>MODIFY DDD
TAG2  DDD    NEW DATA WITH RETURN TERMINATOR  modified line
>P
TAG2  DDD    NEW DATA WITH RETURN TERMINATOR  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 line
>P
TAG3  GGG    NEW DATA WITH ALT MODE TERMINATOR  new data
TAG3  GGG    NEW DATA WITH ALT MODE TERMINATOR  /LINE 7 new form
>B
      .END
>CLOSE
EDITOR XVN Vxxxxx
>

```

Edit Mode Commands

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 a non-directoried output device.

4.6.2 PCLOSE Command

Form:

PCLOSE ↵

Description:

Closes the output file opened by PUT. A PCLOSE will be simulated by the following commands:

TOP
CLOSE
SCLOSE
EXIT

Edit Mode Commands

After any of the above commands, the entire file from the current line at the time to the last line of the file (inclusive) will also be placed in the PUT file. After the last line is copied the PUT file will be closed. If PCLOSE is used to close the PUT file, the current line is not included in the PUT file.

4.6.3 SCLOSE Command

Form:

```
SCLOSE ↵
```

Description:

This command permits the placement of a file onto the current output device. It is particularly useful in closing long files which have only minor changes since the .DAT -15 to .DAT -14 recopying will not take place if .DAT -15 is the current output device.

4.6.4 EXIT [E] Command

Form:

```
EXIT filename ↵
```

NOTE

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

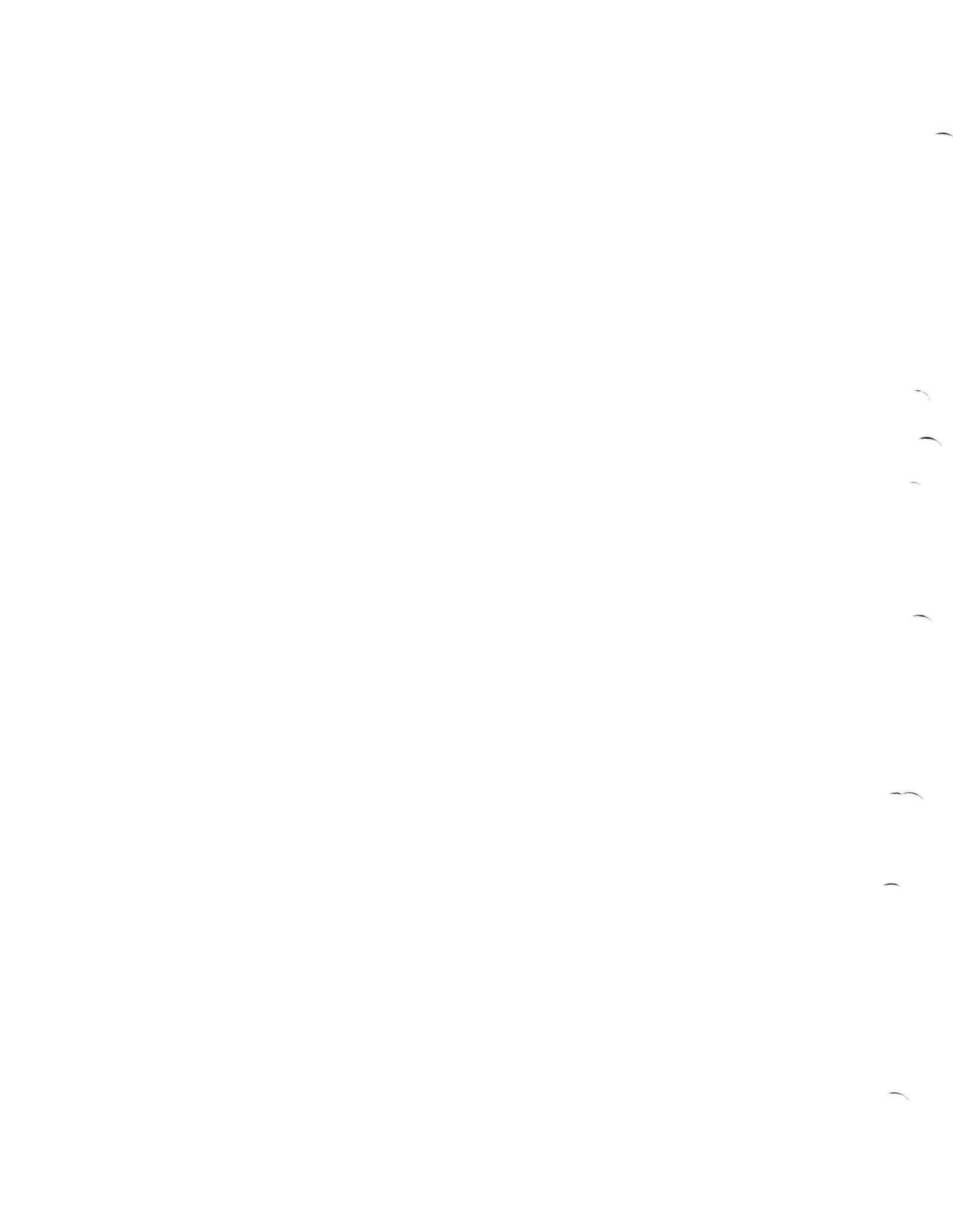
Description:

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.

Example:

```
EDITOR XVM Vnxnnn
>OPEN FILEA SRC
EDIT
>B
BBBB
>EXIT

XVM/DOS Vnxnnn
$
```



APPENDIX A

SUMMARY OF EDITING COMMANDS

<u>COMMAND</u>	<u>ABBREVIATION</u>	<u>OPERATION</u>	<u>PARAGRAPH REFERENCE</u>
APPEND	A	Add string at the rightmost end of the current line.	4.5.9
BLOCK $\left\{ \begin{array}{l} \text{ON} \\ \text{OFF} \end{array} \right\}$	none	Set program to operate in block mode (ON) or in line-by-line mode (OFF).	4.2.5
BOTTOM	B	Make last line of file/buffer the current line.	4.4.3
BRIEF $\left\{ \begin{array}{l} \text{ON} \\ \text{OFF} \end{array} \right\}$	none	Set brief mode to print truncated (ON) or full (OFF) lines.	4.2.8
CHANGE	C	Replace, in the current line, the first occurrence of string1 with string2.	4.5.7
CLOSE	none	Terminate editing on input file and return to the Editor.	4.6.1
CONVERT	none	Change all occurrences of string1 within the file/buffer to string2.	4.5.11
DELETE	D	Delete the current n lines.	4.5.3
EXIT	none	Close the output file and transfer control to Monitor.	4.6.4
FIND	F	Make the first line beginning with string the current line.	4.4.4
GET	G	Add lines from subsidiary input device after (below) the current line.	4.3.4
INSERT	I	Change mode to input.	4.5.5
INSERT string	I	Add string, as a complete line, to the file after (below) the current line.	4.5.6
KEEP	K	Keep (i.e., preserve) the original file (on .DAT -14) for backup purposes.	4.2.10

Summary of Editing Commands

<u>COMMAND</u>	<u>ABBREVIATION</u>	<u>OPERATION</u>	<u>PARAGRAPH REFERENCE</u>
LC (Line Convert)	LC	Same as CONVERT command except that the number of lines to be converted is specified in the command.	4.5.12
LIST	none	Print the current line on the listing device.	4.5.2
LOCATE	L	Make the first line containing string the current line.	4.4.5
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.	4.5.13
MOVE	none	Move a specified number of lines including the current line, to a position in the file/buffer immediately before (above) a line identified by a given string.	4.5.10
NEXT	N	Skip the next n consecutive lines to the new current line.	4.4.2
OPEN	none	Prepare file on input device for editing.	4.2.3
OUTPUT	none	Output selected portions of file.	4.2.9
OVERLAY	O	Replace multiple lines.	4.5.8
PARITY $\left\{ \begin{array}{l} \text{ON} \\ \text{OFF} \end{array} \right\}$	none	Set parity checking to ON or OFF.	4.2.11
PCLOSE	none	Closes the output file opened by PUT.	4.6.2
PRINT	P	Print the current n lines on the console device.	4.5.1
PUT	none	Place portions of a file into many separate source files.	4.2.4
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.	4.5.4
SCLOSE	none	Close file and leave on current output device.	4.6.3
SIZE	S	Set total number of lines in block buffer.	4.2.6

Summary of Editing Commands

<u>COMMAND</u>	<u>ABBREVIATION</u>	<u>OPERATION</u>	<u>PARAGRAPH REFERENCE</u>
TOP	T	Reset current line pointer to beginning of file or buffer.	4.4.1
TV { ON } { OFF }		Enables/Disable soft display when using EDITVP/EDITVT XVM.	Appen- dices B & C
VERIFY { ON } { OFF }	V	Set VERIFY mode to print (ON) or ignore printing (OFF) lines after processing CHANGE, LOCATE, and FIND requests.	4.2.7
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 XVM/DOS systems which have a VP15A Storage Tube Display Unit and the device handler (VPA) 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 addition:

$$\text{TV} \left\{ \begin{array}{l} \text{ON} \\ \text{OFF} \end{array} \right.$$

This command enables or disables the display according to the ON/OFF parameter. The command may be given at any time. Typing TV ON implies VERIFY OFF (no Teletype output); however, typing TV OFF does not automatically set VERIFY ON. When EDITVP is first loaded, TV is OFF. If, at the top of a file or BLOCK (i.e., OPEN or TOP was just typed), TV ON is typed, a Locative Request (e.g., NEXT, FIND, LOCATE, etc.) must be given before any lines are displayed. If TV ON is typed after the execution of a Locative Request, a group of lines most recently added to the output file or block will be displayed - with the current line (to which modifications apply) appearing as the last line.

EDITVP uses the same .DAT slot assignments as the Editor with the addition of .DAT slot 10 which must be assigned to VPA (via the ASSIGN command). EDITVP is supplied as a system program and is requested by the following XVM/DOS command:

\$EDITVP ↵

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 and the modified line is displayed below it. When a section of text which has been modified as described above is re-displayed (after a TOP command) the crossed-out line is eliminated.

EDITVP

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 CRT 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 the VP15A.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 file. 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.

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

EDITVP

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.

- a) Announcements, error messages and ">" symbols are echoed on the console printer.
- 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



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.
- c) \$↑X (control X) Turn on VT display unit.

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.

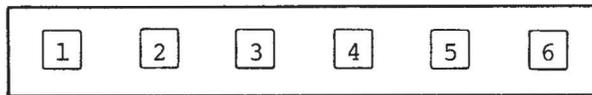
EDITVT

VT15 CONTROLS AND DISPLAY MODES

CONTROLS

The VT15 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 numbers 5 and 6, based on the following numbering scheme:



DISPLAY MODES

The VT15 Display operates in two display modes:

- a) SCROLL Mode - When the number 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 number 5 pushbutton is in 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. When a large file is to be displayed in the PAGE mode, the number 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.

WARNING

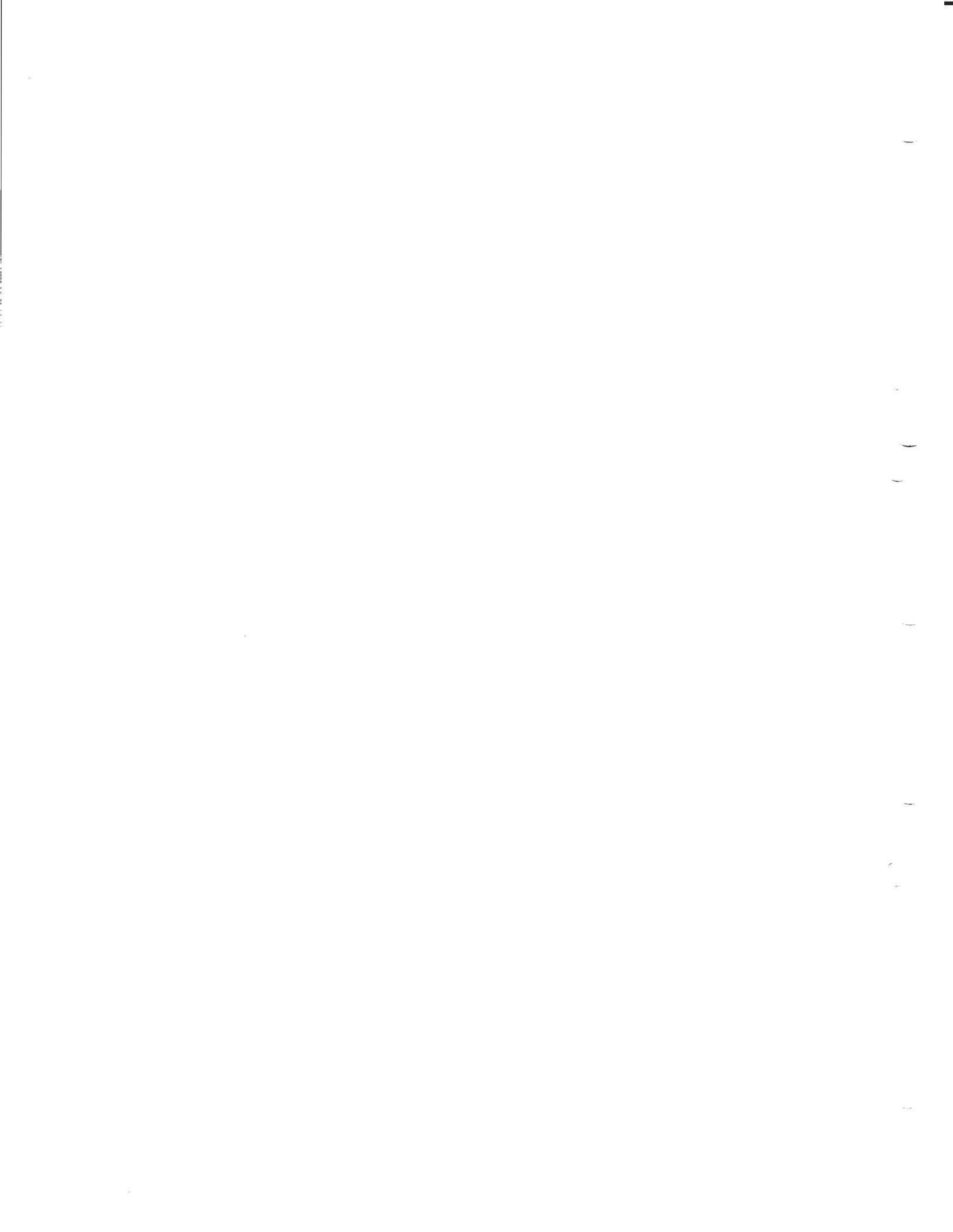
The EDITOR may appear hung when number 5 is ON (lit) and the page is full (i.e., waiting for change in state of number 6).

EDITVT

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.
- 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 the screen will clear when the next output is made to the VT15 (e.g., new line added, mode change).



INDEX

- Abbreviated commands, 4-1
- Add data, 4-31
 - APPEND command, 4-31
 - GET command, 4-16
- ALT MODE, 3-2, 4-40
- APPEND command, 4-31
- ASSIGN command, 2-4
- @ (at symbol) usage, 2-7

- Blanks in commands, 4-1
- Block mode, 2-1, 2-2
 - buffer overflow, 3-6
 - edit, 3-3
- BLOCK OFF command, 3-3, 4-5
- BLOCK ON command, 3-3, 4-5, 4-8
- BOTTOM command, 4-20
- Brackets [] used as symbol for optional command string elements, 1-2
- BRIEF command, 4-9
- Brightening, B-2
- Buffer,
 - overflow in block mode, 3-6
 - search, 2-2
 - size, 4-5, 4-7
 - space, 4-35

- CALL DELETE command, 4-3
- CALL RENAME command, 4-2
- Calling errors, 3-6
- Carriage RETURN, 3-2
- Carriage RETURN-LINE FEED symbol (↵), 1-2
- CHANGE command, 4-29
- Change items in file, 4-37
- Character deletion, 2-6
- Checksum, 3-5
- Clear screen commands, B-3, C-3
- CLOSE command, 3-2, 3-3, 4-43, B-3, C-3
- Closing the new file, 3-3
- Command abbreviation, 4-1
- Command mode, 2-1
- Command string errors, 3-4

- Commands,
 - clear screen, B-3, C-3
 - edit close operation, 4-43
 - input/output, 4-13
 - line pointer control, 4-17
 - locative, 4-17
 - manipulative, 4-22
 - text modification, 4-22

- Commands,
 - APPEND, 4-31
 - BLOCK OFF, 3-3, 4-5
 - BLOCK ON, 3-3, 4-5, 4-8
 - BOTTOM, 4-20
 - BRIEF, 4-9
 - CALL DELETE, 4-3
 - CALL RENAME, 4-2
 - CHANGE, 4-29
 - CLOSE, 3-2, 3-3, 4-43, B-3, C-3
 - CONVERT, 4-36, B-3, C-3
 - DELETE, 4-25, B-3, C-3
 - EDITVP, B-1
 - EDITVT, C-1
 - EXIT, 4-44, B-3, C-3
 - FIND, 4-20
 - GET, 4-16, B-3, C-3
 - \$HALF ON/OFF, C-1
 - INSERT, 3-2, 4-27
 - INSERT string, 4-28
 - KEEP, 4-12
 - LC (Line Convert), 4-38, B-3, C-3
 - LIST, 4-24
 - LOCATE, 4-21
 - MODIFY, 4-40
 - MOVE, 4-32
 - NEXT, 4-18
 - OPEN, 3-1, 3-3, 4-4
 - OUTPUT, 4-10
 - OVERLAY, 4-30
 - PARITY, 4-13
 - PCLOSE, 4-43
 - PRINT, 4-22, B-3, C-3
 - PUT, 4-5
 - READ, 4-14
 - RENEW, 4-6, 4-15
 - RETYPE, 4-26
 - SCLOSE, 4-44
 - SIZE, 3-3, 4-7

INDEX (Cont.)

- Commands (cont.),
 summary, A-1
 TOP, 4-18, B-3, C-3
 TV ON/OFF, B-1, B-3, C-3
 VERIFY, 4-8
 \$VT ON, C-1
 WRITE, 3-3, 4-15
 \$↑X, C-1
 Control modes, 2-1, 3-2, B-2, C-2
 CONVERT command, 4-36, B-3, C-3
 Creating a new file, 3-2
 CTRL P keyboard command, 2-6,
 4-40
 CTRL U, 2-7, 4-40
- Dashed line on display unit, B-1
 Data files, 2-3
 Data modes, 2-1
 DAT slot assignments, 2-4, B-1
 DELETE command, 4-25, B-3, C-3
 Deletion of
 characters, 2-6
 line, 2-7
 Device assignment table (.DAT), 2-4
 Directoried device, 2-4
 Display controls, VP15A, B-2
 Display modes,
 VP15A, B-2
 VT15, C-2
 Duplicate file names, 3-7
- Edit block mode, 3-3
 Edit close operation commands,
 4-43
 Edit mode, 2-1, 3-2
 Edit mode commands, 4-1
 EDITVP, B-1
 EDITVT, C-1
 End-of-file, premature, 3-4
 Erase
 characters, 2-6
 line of typing, 2-7
 ERASE push button, B-2, B-3
 Error-handling conventions, 3-4
 Example file, 4-2
 EXIT command, 4-44, B-3, C-3
- File,
 closing, 3-3, 4-43, 4-44
 creation, 3-2
 deletion, 3-8, 4-3
- File (cont.),
 example, 4-2
 modification, 3-1
 naming errors, 3-6
 opening, 4-4
 renaming, 3-8, 4-2
 saving, 4-12
 Filename ext, 3-1
 Filenames, duplicate, 3-7
 Files,
 input, 2-4
 subsidiary, 2-4
 FIND command, 4-20
- GET command, 4-16, B-3, C-3
- \$HALF ON/OFF command, C-1
- INPUT control mode, 2-1
 Input files, 2-4
 Input line truncation, 3-6
 INPUT mode, 3-2
 Input/output commands, 4-13
 Input/output, monitor, 2-3
 INSERT command, 3-2, 4-27
 INSERT line command, 4-28
- KEEP command, 4-12
- LC (Line Convert) command, 4-38,
 B-3, C-3
- Line,
 deletion, 2-7, 4-25
 insertion, 4-27, 4-28
 modification, 4-40
 overflow, 3-5
 processing, 2-3
 replacement, 4-26, 4-30
 Line-by-line data mode, 2-1
 Line-by-line search, 4-20
 Line pointer, 2-1, 4-15
 control commands, 4-17
 Line printer, Unichannel, 4-24
 LIST command, 4-24
 LOCATE command, 4-21
 Locative commands, 4-17

INDEX (Cont.)

Manipulative commands, 4-22
Modes of operation, 2-1, 3-2,
4-5
MODIFY command, 4-40
Modifying an existing file, 3-1
Monitor I/O, 2-3
MOVE command, 4-32

NEXT command, 4-18
Nondirectoryed device, 2-4

OPEN command, 3-1, 3-3, 4-4
OUTPUT command, 4-10
Output file conventions, 2-6
Output files, 2-5
Overflow, block mode buffer, 3-6
OVERLAY command, 4-30

Page mode,
VP15A, B-3
VT15, C-2
Parity, 3-5
PARITY command, 4-13
PCLOSE command, 4-43
Pointer,
see Line pointer
Premature end-of-file, 3-4
PRINT command, 4-22, B-3, C-3
Push buttons,
VP15A, B-3
VT15, C-3
PUT command, 4-5

READ command, 4-14
Read errors, 3-5
RENEW command, 4-6, 4-15
Replace current line 4-26
Replace lines, 4-30
Restart Editor, 2-6
RETYPE command, 4-26
RUBOUT, 2-6, 4-40

Save current file, 4-12
Scan mode, VP15A, B-2
SCLOSE command, 4-44
Scratch device, 2-5
Scroll mode, VT15, C-2

Searching, 2-2, 4-21
Setup commands, 4-2
EDITVT, C-1
SIZE command, 3-3, 4-7
Spaces in commands, 4-1
Space symbol (␣), 1-2
Storage tube display unit, VP15A,
B-1
Subsidiary files, 2-4
Summary editing commands, A-1
Symbols used in this manual, 1-2

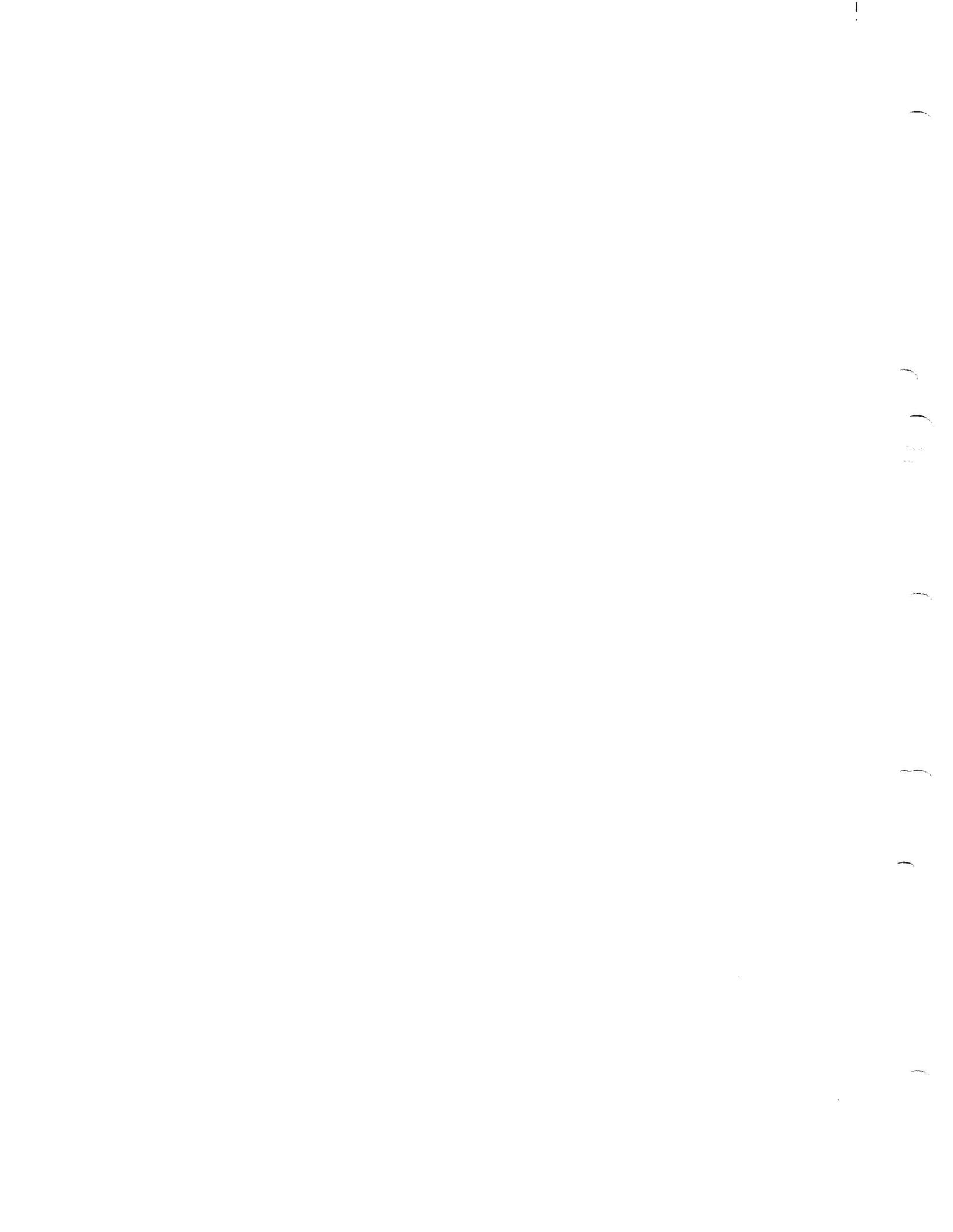
Tab symbol (→), 1-2
Terminate editor output, 2-7
Text mode, 2-1
Text modification, 2-2
commands, 4-22
TOP command, 4-18, B-3, C-3
Truncation, 3-6, 4-29, 4-37, 4-40
TV ON/OFF command, B-1, B-3, C-3

UFDT slot assignments, 2-4
Unichannel line printer, 4-24

VERIFY command, 4-8
Verify mode, 4-10
VIEW push button, B-2
VP15A storage tube display unit,
B-1
VT15 controls and display modes,
C-2
VT15 graphics display unit, C-1
\$VT ON command, C-1

WRITE command, 3-3, 4-15

\$↑X command, C-1



READER'S COMMENTS

NOTE: This form is for document comments only. Problems with software should be reported on a Software Problem Report (SPR) form.

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

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Is there sufficient documentation on associated system programs required for use of the software described in this manual? If not, what material is missing and where should it be placed?

Please indicate the type of user/reader that you most nearly represent.

- Assembly language programmer
- Higher-level language programmer
- Occasional programmer (experienced)
- User with little programming experience
- Student programmer
- Non-programmer interested in computer concepts and capabilities

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code _____

or
Country

If you require a written reply, please check here.

Please cut along this line.

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

Software Communications
P. O. Box F
Maynard, Massachusetts 01754



digital

digital equipment corporation